VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

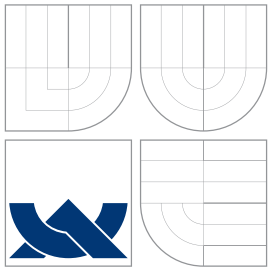
ZMĚNA RYCHLOSTI ŘEČI

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

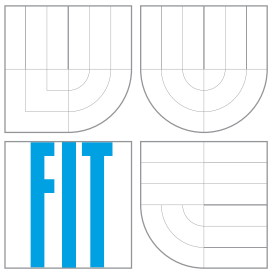
AUTOR PRÁCE
AUTHOR

ALEŠ KOVÁŘÍK

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ZMĚNA RYCHLOSTI ŘEČI

MODIFICATION OF SPEECH RATE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — ALEŠ KOVÁŘÍK
AUTHOR

VEDOUCÍ PRÁCE — Ing. IGOR SZÖKE
SUPERVISOR

BRNO 2007

Zadání diplomové práce

Řešitel:	**Kovářík Aleš**
Obor:	Výpočetní technika a informatika
Téma:	**Změna rychlosti řeči**
Kategorie:	Signály a systémy

Pokyny:

1. Seznamte se s teorií změny rychlosti řeči (zejména metoda PSOLA).
2. Implementujte vybranou metodu v C++.
3. Proveďte experimenty a určete kvalitu řeči generované aplikací.
4. Zlepšete kvalitu řeči pomocí integrace fonémového rozpoznávače.
5. Porovnejte dosažené výsledky a navrhněte další postupy.

Implementační jazyk:	C++
Operační systém:	MS Windows nebo Linux

Vedoucí:	Szöke Igor, Ing., UPGM FIT VUT
Datum zadání:	1. listopadu 2006
Datum odevzdání:	22. května 2007

Licenční smlouva

Licenční smlouva je uložena v archivu Fakulty informačních technololgí Vysokého učení technického v Brně.

Abstrakt

Tato diplomová práce pojednává o změně rychlosti řeči. Jako metoda pro změnu rychlosti byla v této práci použita PSOLA (Pitch Synchronous OverLap Add). PSOLA je algoritmus pracující v časové oblasti. Práce rovněž uvádí další metodu – fázový vokodér pracující ve frekvenční oblasti. Tato práce dále rozšiřuje metodu PSOLA o fonémový rozpoznávač s úmyslem zvýšit srozumitelnost výsledné řeči vyhodnocením typu hlásek v mluveném projevu – samohlásky, frikativy apod. Pro ověření kvalit navrženého propojení metody PSOLA a fonémového rozpoznávače byla vytvořena aplikace, která toto spojení realizuje.

Klíčová slova

řeč, rychlost, PSOLA, časová oblast, fonémový rozpoznávač

Abstract

This diploma thesis discusses modification of a speech rate. The PSOLA (Pitch Synchronous OverLap Add) method was used for the rate modification. This algorithm works in time domain. Another method – phase vocoder, which works in frequency domain is also presented in an overview. This thesis extends the PSOLA method with a phoneme recognition, which allows for better understandability of the speech output by considering characteristics of the phonemes beeing pronounced. To examine this proposed method, an application connecting PSOLA and a phoneme recognizer was developed.

Keywords

speech, speed, rate, PSOLA, time domain, phoneme recongizer

Citace

Aleš Kovářík: Změna rychlosti řeči, diplomová práce, Brno, FIT VUT v Brně, 2007

Změna rychlosti řeči

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Igora Szökeho

.....
Aleš Kovářík
19. května 2007

Obsah

Kapitola 1

Úvod

Při mezilidské komunikaci stále hraje dominantní roli verbální komunikace. Při konverzaci dvou či více lidí přijímáme proud jejich řeči v takové rychlosti jakou jej hovořící osoba vyslovuje. Někdy to je skutečně rychlé tempo, ve kterém se přestáváme orientovat jindy je to tempo velmi pomalé, které naopak unavuje. Při osobním kontaktu můžeme zpětnou vazbou naznačit, že by měl řečník změnit rychlost svého projevu. Ovšem současnost posluchače často staví do pozice pasivní, jelikož k nám řeč přichází skrze elektronická média.

Máme-li však hlas v digitální podobě, můžeme si pomoci i v takovémto případě a tempo řeči můžeme změnit. Úprava rychlosti řeči rovněž najde uplatnění v komerčních médiích. Zadavatel reklamy do rádia či televize potřebuje vtěsnat co nejvíce informací do co nejkratšího časového úseku, jelikož platí, že čas jsou peníze. Typickým příkladem je informace o lécích, jež nabádá, aby uživatelé těchto léků konzultovali vhodnost užití se svým doktorem či lékárníkem. Samotné doporučení konzultace s lékařem lék neprodá, ale je vyžadováno zákonem. Snaží-li se herec namlouvající reklamu mluvit rychle, může to působit spíše směšně. Zvýšení rychlosti se pak řeší elektronicky, přičemž je však nutné zachovat zvukovou kvalitu hlasového projevu tak, aby nezněl jako postavička z kreslených seriálů. V tomto okamžiku nastupuje některá ze sofistikovaných metod sloužící pro změnu rychlosti řeči při zachování jejího tónu.

Hlasový projev se nejprve snímá a digitalizuje do počítače, čímž se získá řada vzorků (samples). Pokud nějaký algoritmus pracuje přímo na této řadě vzorků, říkáme, že pracuje v časové oblasti, pokud nejdříve převede signál na jeho frekvenční složky pomocí Fourierovy transformace a dále pracuje s frekvencemi, mluvíme o zpracování signálu ve frekvenční oblasti. V této práci představím dvojici metod pro změnu rychlosti řeči v časové a frekvenční oblasti. Detailně pak rozvedu metodu PSOLA (Pitch Synchronous OverLap Add). Pro tuto metodu je velmi důležité zjistit základní tón hlasu mluvčího. K tomuto účelu rovněž slouží celá řada algoritmů, část této práce jim tedy budu rovněž věnovat.

Metoda PSOLA sama o sobě produkuje kvalitní výstup, přesto však je možné pokusit se dosáhnout zvýšení srozumitelnosti hlasového výstupu, tak že přihlédneme k typu vyslovovaných hlásek. Je totiž snadné prodloužit či zkrátit znělou samohlásku jako je *a* nebo *e*, ale horší to už může být se souhláskami jako jsou *t* nebo *c*. Z uvedeného plyne, že je potřeba předpřipravit půdu metodě PSOLA další komponentou, která jí určí typ hlásky. V mé práci k tomu slouží fonémový rozpoznávač. Popisu jeho integrace do celého řetězce zpracování řečového signálu se budu věnovat v 5. kapitole této práce.

Tato práce nenavazuje na můj ročníkový projekt PI1/PI2 ani na jiný diplomový projekt.

1.1 Cíle diplomové práce

Cílem této práce je vytvořit aplikaci, která bude metodou PSOLA měnit délku řečového signálu bez vlivu na tón jakým mluvčí hovoří. Implementace PSOLy bude modulární a umožní její snadné vložení do jiných aplikací. PSOLA vyžaduje přesné určení základního tónu řeči, a tak bude uvedeno několik algoritmů, z nichž některé budou implementovány v ukázkové aplikaci. Samotná implementace PSOLy bude umožňovat snadnou náhradu tohoto algoritmu. Současně s metodou PSOLA bude realizován jednoduchý algoritmus pro změnu délky signálu, který bude sloužit k porovnání kvalit hlasového výstupu. V další části bude metoda PSOLA rozšířena o rozpoznávání fonémů, od čehož se bude očekávat především zvýšení srozumitelnosti hlasového výstupu vytvořené aplikace.

Kapitola 2

Charakter řečového signálu

Dříve než se začnu věnovat samotnému zpracování řečového signálu, je nutné pochopit, v čem se řečový signál odlišuje od jiných. Ze způsobu jakým se tvoří v řečovém ústrojí člověka vyplyne, jaké jeho vlastnosti jsou podstatné pro jeho zpracování v kontextu této diplomové práce.

2.1 Tvorba řeči

Tvorba řeči v lidském hlasovém ústrojí je velmi komplexní proces, který vyžaduje přesnou synchronizaci mnoha svalů od pobřišnice a mezižeberních svalů, které řídí dýchání, po svaly v hlasivkách, hrtanu, jazyku, rtů, svalů pohybujících čelistí a dalších.

Hnací silou tvorby řeči je proud vzduchu, který vzniká při dýchání. Čeština využívá pro tvorbu hlasu pouze fáze výdechu, jsou ovšem i jazyky (například některé africké), které využívají k tvorbě hlasu i nádech. Při patřičně nastaveném řečovém ústrojí se začne proud vzduchu vytvářeného plícemi formovat do podoby řeči. Tento proud po opuštění plicních laloků naráží v hrtanu na překážku v podobě hlasivek. Hlasivky jsou jakousi membránou, či chlopní, která má napříč pružnou štěrbinu. Při normálním dýchání je štěrbina hlasivek roztažená a vzduch jí volně prochází, avšak v hovoru se zúží či zcela uzavře a vytvoří tak podmínky pro vznik zvukových vln. Tvar štěrbiny se při mluvení mění. Ovládání hlasivkové štěrbiny je podvědomé, ale je možné jej ovládat i vědomě. Na příklad při učení se cizímu jazyku se učíme i vyslovovat některé nové hlásky, která v češtině nejsou.

Při patřičném stažení hlasivkové štěrbiny se pod ní díky vzduchu vytlačovanému z plic zvyšuje tlak, který ji při dosažení jisté úrovně otevře, následkem čehož se sníží tlak pod hlasivkami a zvýší tlak nad nimi. Štěrbina se znovu uzavře a proces se opakuje. Tento jev je velmi rychlý – například při zpěvu tónu *a* vykonávají hlasivky 435 kmitů za vteřinu [3]. Výsledkem kmitajících hlasivek je série rychlých změn tlaku, a dochází tak k vytvoření **základního tónu hlasu**.

Tento hlas bez dalšího zpracování v řečovém ústrojí zní poměrně slabě a řezaně. Zvukové vlny vytvářené hlasivkami dále procházejí rezonančními dutinami. Tyto dutiny jsou: horní část dutiny hrtanové, hltanová, dutina ústní, nosohltanová a nosní. V nich díky rezonanci základní tón zesílí a je obohacen o vyšší harmonické složky, čímž získá charakter lidského hlasu. Řečový signál však neutváří jen rezonanční dutiny. Důležité jsou například rty a jazyk, které svým postavením jednak upravují tvar rezonančních dutin (jazyk mění tvar a objem dutiny ústní), ale také nastavují překážky proudu vzduchu, při jejichž uvolnění vznikají některé souhlásky jako třeba *p* nebo *t*.

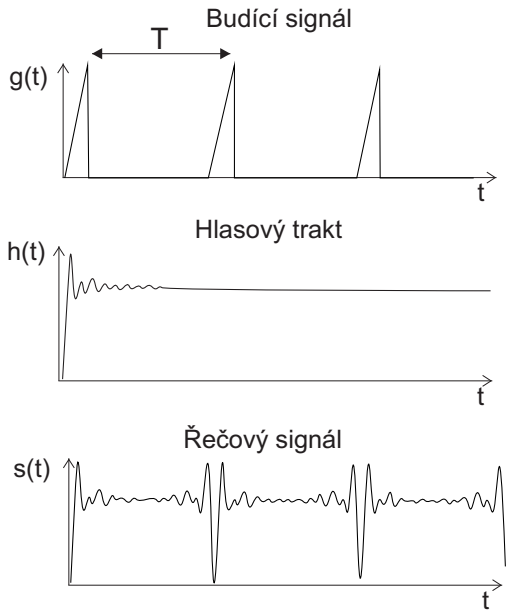
Z uvedeného plyne, že tvorba řeči se dá rozdělit do třech částí, jež se dají modelovat prostředky známými z teorie signálů.

- Plíce → *zdroj energie*
- Hlasivky → *buzení*
- Rezonanční dutiny, jazyk, rty atd. → *filtr*

Buzení produkuje řadu impulzů, které vzbuzují impulzní odezvu ve filtru. Při konstantní konfiguraci řečového ústrojí je výsledný signál $s(t)$ tvořen konvolucí buzení $g(t)$ a impulzní odezvy filtru $h(t)$:

$$s(t) = g(t) * h(t) \tag{2.1}$$

Na obrázku 2.1 je tato konvoluce graficky znázorněna.



Obrázek 2.1: Princip tvorby znělé hlásky v časové oblasti

Hlasivky v průběhu řeči nevytvářejí základní tón neustále. Jednak člověk do řeči vkládá pauzy mezi slovy, ale především ne všechny hlásky, resp. fonémy[1], hlasivky využívají. Fonémy, při jejichž vyslovování hlasivky nekmitají se nazývají neznělé. Fonémy, které hlasivky využívají se nazývají znělé. Mezi neznělé fonémy, patří např. ‘s’, ‘ch’ a další, při nichž je hlasivková štěrbina roztažena téměř do klidového stavu a tření proudu vzduchu o její hrany vytváří pouze šum. Vliv rezonančních dutin je v takovém případě slabší a významnými se stávají různé překážky vzduchu jako jsou zuby, jazyk či rty.

[1] Foném je nejmenší jednotkou řeči, která je významotvorná, tj. může rozlišovat jednotlivá slova s různým významem. [9]

2.2 Rychlost řeči

Víme-li nyní jak vzniká řeč můžeme se zamyslet nad tím, která část hlasového ústrojí mění rychlost jakou hovoříme. Plíce jako zdroj energie rychlost neovlivňují vůbec – více energie dodané plícemi pouze zvýší hlasitost projevu. Hlasivky svým kmitáním udávají základní tón řeči, takže zvýšíme-li frekvenci jejich kmitání, zvýšíme pouze frekvenci základního tónu řeči, ale mluvit budeme stále stejně rychle. Zbývají již jen orgány artikulačního ústrojí. Každému písmenu, resp. fonému, které vyslovujeme odpovídá jistá konfigurace tohoto ústrojí a čím rychleji tuto konfiguraci měníme, tím rychleji mluvíme. Toto je důležitý poznatek pro tento projekt. Chceme-li zachovat výšku hlasu, musíme při změně rychlosti řeči upravovat pouze konfiguraci hlasového ústrojí. Vezměme si pro příklad libovolnou znělou samohlásku. Její záznam vypadá jako na obrázku 2.1 $s(t)$. Vliv artikulačního ústrojí se projevuje v úseku mezi tzv. excitacemi (okamžiky buzení). Tento úsek se nazývá perioda základního tónu (Anglicky pitch period) a maximum na jejím počátku je tzv. pitch mark (český ekvivalent tohoto názvu zřejmě neexistuje). Pro zachování charakteru mluvčího tedy musíme při změně rychlosti řeči tyto úseky zachovat v maximální možné míře nezměněné.

Kapitola 3

Přehled číslicového zpracování řeči

Digitální zpracování akustických signálů, mezi které řečový signál patří, má zatím nedlouhou historii. Hovoříme-li o skutečném zpracování signálu, tedy ne jen o jeho záznamu a přenosu, které umožňovalo kupříkladu rádio či televize již začátkem minulého století, pak skutečné úpravy signálu začaly být realitou až v druhé polovině 20. století. Ve svých počátcích se jednalo o zpracování analogové. Složité obvody filtrovaly analogový signál, avšak možnosti byly omezené zvláště srovnáme-li je se současností. Možnosti práce s řečí pak už byly vskutku minimální. S příchodem elektronických digitálních počítačů se odborníci začali zajímat o možnost jejich využití. Později, s nárůstem výkonu domácích počítačů, se začali možností manipulace s audio nahrávkami zabývat i běžní uživatelé. Principy číslicového zpracování audio signálů a především signálů obsahujících mluvený projev se bude zabývat tato kapitola.

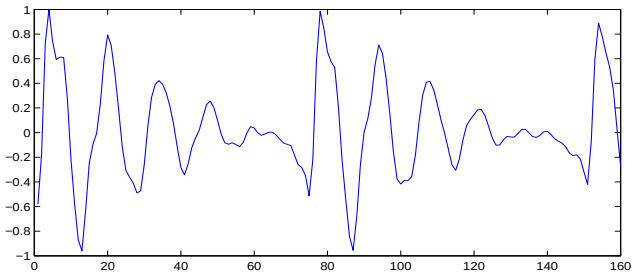
3.1 Hlasový záznam jako diskrétní signál

Hlasový signál je v počítači reprezentován jako jakýkoliv jiný akustický signál. Na počátku je nutné hlasový projev zaznamenat, tedy digitalizovat. Při zpracování řeči si ve většině případů vystačíme s jedním audio kanálem. Digitalizace převádí spojitý signál na řadu diskrétních hodnot. Tento převod probíhá tak, že se v pravidelných intervalech zaznamená vzorek signálu a jeho hodnota se kvantizuje. Rychlost vzorkování se nazývá vzorkovací frekvence (anglicky sample rate). Pro zpracování řeči jsou vhodné vzorkovací frekvence od 8000 Hz, při které se dle Nyquistova teorému zachytí frekvence akustického signálu od 0 do 4000 Hz. S rostoucí vzorkovací frekvencí stoupá kvalita zaznamenaného signálu/řeči, ale i náročnost následných výpočtů, a při volbě frekvence je tedy vhodné zvážit, k jakému účelu budou výstupy používány. Pro většinu algoritmů, které řeč analyzují postačuje frekvence 8000 Hz.

3.2 Metody práce s hlasovým signálem

Máme-li mluvený projev zaznamenán v digitální podobě, můžeme s ním začít manipulovat. Řečový signál si už však žádá specifické zacházení. Z pohledu teorie signálů je řeč náhodný signál. Pro určení jeho parametrů však potřebujeme, aby byl stacionární.[1] Vycházíme z předpokladu, že řečový signál se mění relativně pomalu, a proto, rozdělíme-li jej na úseky dostatečně krátké, můžeme tyto považovat za stacionární. Těmto krátkým

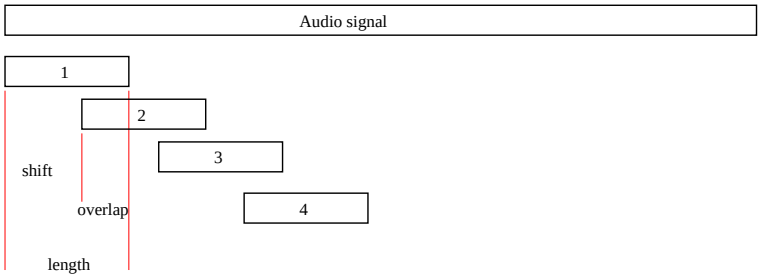
[1] Stacionární signál – charakteristické veličiny se nemění s časem.



Obrázek 3.1: Řečový rámec délky 20 ms při vzorkovací frekvenci 8000 Hz

úsekům řečového signálu říkáme hlasové rámce. Hlasové rámce se volí s délkou (length) takovou, aby zachytila alespoň jednu periodu základního tónu lidského hlasu. Základní tón mužského hlasu se pohybuje v rozmezí přibližně 70–120 Hz. Ženský hlas má základní tón v rozsahu 150–300 Hz a u dětí dosahuje až frekvence 400 Hz [12]. Nejnižší tón je tedy kolem 70 Hz. Pro tuto frekvenci musí mít rámec délku nejméně $\frac{1}{70Hz} = 14{,}3$ ms. V praxi se používají rámce délky 20 ms ale i více – v závislosti na konkrétních aplikačních požadavcích.

Jak již bylo řečeno, mění se hlasový signál relativně pomalu, pokud ale budeme signál dělit na rámce (segmentovat) jeden za druhým, můžou se nám parametry vzešlé z takovýchto rámců poměrně skokově měnit. Z tohoto důvodu volíme ještě překryv (overlap, potažmo posun – shift) mezi sousedními rámci. Čím větší překryv rámců zvolíme, tím vyhlazenější průběh parametrů tak získáme. Velký překryv však zvyšuje výpočetní náročnost zpracování signálu, jelikož se větší část signálu zpracovává vícenásobně. Pro ilustraci, posun roven délce signálu znamená žádný překryv a každý vzorek se analyzuje jednou. Při posunu rovném polovině délky rámce již všechny vzorky (s výjimkou začátku a konce signálu) projdou analýzou dvakrát. Při posunu o třetinu rámce – tedy překrytí 2/3, jsou vzorky analyzovány třikrát. Opět rozhodnutí není univerzální, ale řídí se konkrétními požadavky dané aplikace. V projektu byly použity rámce délky 20 ms a překryv 10 ms. Ilustrace segmentace řečového signálu je na obrázku 3.2.



Obrázek 3.2: Rozdělení audio signálu na rámce

Metody mající za úkol změnu řečového signálu umožňují kromě změny rychlosti řeči také upravovat tón, jakým řečník hovoří. Existují například i metody pro sloučení dvou různých mluvčích v jeden nový hlas, což bylo použito ve filmu Farinelli, kde byl vytvořen zpěv kastráta sloučením zpěvu sopranistky Ewy Mallas-Godlewské a altového zpěváka Dereka Lee Ragina. Další zajímavou metodou je změna hlasu mluvčího tak, aby se podobal jinému

člověku[11], což je možné provést jak v časové tak i frekvenční oblasti.

3.2.1 Předzpracování audio signálu

Při analýze řeči nás někdy zajímají jen její části nacházející se v konkrétním frekvenčním pásmu. Tehdy je vhodné použít relativně jednoduché filtry, která umožní ořezávat vysoké frekvence, či potlačovat nízké. K tomu nám mohou posloužit různé filtry typu FIR (Finite Impulse Response) nebo IIR (Infinite Impuse Response). Použití takovýchto algoritmů, které přímo neanalyzují řečové charakteristiky, ale pouze připravují půdu sofistikovanějším algoritmům, se říká preprocessing:

- **Frekvenční propusť** se často používá pro předpřípravu signálu, než z něj začneme extrahovat parametry řeči. *Dolní propusť* se aplikuje před zjišťováním základního tónu. Má-li základní tón frekvenci v rozmezí zhruba 70–400 Hz, pak dolní propustí se vybere oblast, řekněme, od 0 do 1000 Hz a algoritmy jako např. autokorelační funkce pak mohou podávat přesnější výsledky. Naopak charakter horní propusti má *preemfáze*. Rozložení energie řečového signálu ve frekvenčním spektru ukazuje, že podstatná část energie leží pod hranicí 300 Hz, přičemž většina užitečné informace z řečového signálu leží nad touto hranicí. Preemfáze tak má za úkol tuto oblast zesílit. Jedná se o jednoduchý FIR filtr prvního řádu 3.1:

  $$H(z) = 1 - \kappa z^{-1} \tag{3.1}$$

  který se v diskrétní časové oblasti vyjádří vztahem:

  $$s'[n] = s[n] - \kappa s[n-1] \tag{3.2}$$

- **Klipování** se používá jako předstupeň autokorelační funkce (sekce 4.1.1). Pro zpracovávaný signál se určí úroveň, na kterou se signál „ořízne“ (clip) – vzorky s hodnotou pod touto úrovní se přepíší na nulu a vzorky nad buď nabudou hodnoty klipovací úrovně anebo si zachovají původní hodnotu. V ideálním případě po oříznutí zbudou pouze vrcholky buzení – tedy počátků period základního tónu a autokorelační funkce tak bude moci podávat lepší výsledky. Výpočet klipovací úrovně je dán vztahem 3.3. Konstanta k se volí 0,6–0,8.

  $$c_L = k \max_{n=0\ldots N-1} |x(n)|, \tag{3.3}$$

3.3 Změna rychlosti řeči

Změnou rychlosti řeči míníme změnu délky projevu, ovšem bez vlivu na tón, jakým řečník hovoří. Pokud se změní délka projevu na polovinu, stále musí být možné rozpoznat hovořícího člověka. Změna rychlosti řeči je vcelku komplexní úkol. Jak bylo uvedeno v druhé kapitole, rychlost řeči závisí na změně v artikulačním ústrojí člověka. Efekt artikulačního ústrojí se však v řečovém signálu projevuje v závislosti na buzení – základním tónu. Pokud bychom netrvali na požadavku zachování kvality řeči, mohli bychom signál jednoduše převzorkovat a přehrát jej na původní vzorkovací frekvenci. V takovém případě se však změní všechny frekvenční složky signálu,výsledkem čehož by se při zkrácení délky signálu zvýšil základní tón a člověk by zněl jako postavička z animovaných seriálů. Opačným extrémem by bylo

prodloužení signálu, kdy by základní tón poklesl. Takový způsob úpravy rychlosti je analogický ke změně otáček gramofonu či kazetového přehrávače. Z jmenovaných důvodů je nutné pro kýžený výsledek signál zpracovat speciálními algoritmy, z nichž některé zde nyní představím.

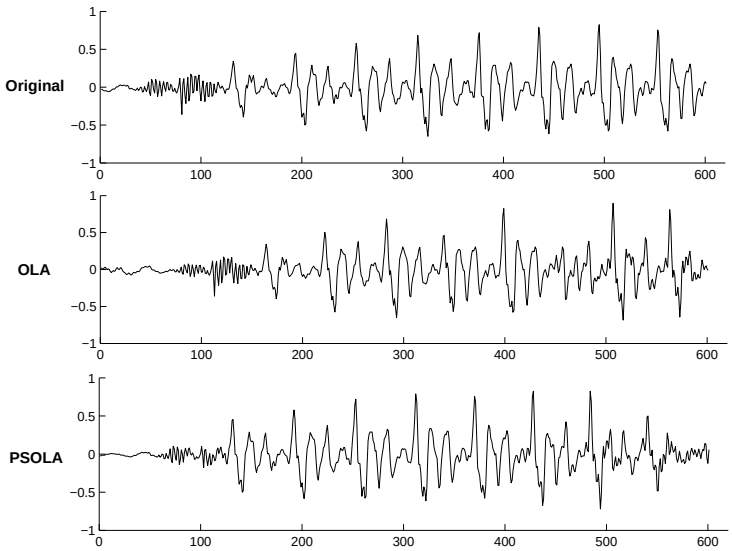
3.3.1 Přehled metod

OLA

Poměrně jednoduchá metoda, která pracuje v časové oblasti. Její název je zkratkou jejího základního principu – OverLap Add. Pomocí okénkovacích funkcí vybírá úseky signálu a pak je přes sebe překrývá a sečítá. Takovým postupem se vyhladí přechody mezi napojovanými segmenty. OLA v zásadě pracuje se stejně dlouhými segmenty, do kterých se podle výšky základního tónu vejde několik jeho period anebo jen část. Tím, že se přes sebe překládají segmenty bez změny jejich délky, zachová se frekvenční složení daného segmentu a tedy charakter hlasu. Ovšem to, že segment neodpovídá přesně periodám základního tónu, způsobuje vytváření zvukových artefaktů. Změny délky signálu se dosáhne tím, že některé segmetny se buď zopakují nebo vypustí. Při použití segmentů, které se svou délkou přibližují periodě základního tónu, funguje OLA uspokojivě, ale řeč není čistá kvůli nenavazujícím periodám základního tónu.

PSOLA

PSOLA je rozšířením metody OLA. Stejně jako OLA pracuje v časové oblasti, přičemž pro svojí funkci potřebuje přesně detekovat základní tón řeči, resp. pitch marky. Se signálem pak pracuje na úrovni period základního tónu, což zajistí, že při odpovídající manipulaci s těmito periodami nedojde ke změně výšky hlasu. Detailní popis PSOLy bude v kapitole 4.



Obrázek 3.3: Výsledky algoritmů OLA a PSOLA pro rychlost 1,5. PSOLA oproti OLe zachovává čisté periody. Patrné je to zejména v oblasti mezi vzorky 400 a 500. PSOLA může mít odobně chybné napojení segmentů, pracuje-li se špatně určenými pitch marky.

Fázový vokodér (Phase vocoder)

Fázový vokodér pracuje ve frekvenční oblasti. Algoritmus se skládá ze třech základních kroků[4]:

1. Provede se STFT – Short Time Fourier Transform (Krátkodobá Fourierova transformace). Vstupní signál se rozdělí na krátké segmenty pomocí okénkovací funkce, kterou může být Hanningova nebo Gaussova funkce. Nad jednotlivými segmenty, které se částečně překrývají se provede rychlá Fourierova transformace FFT.

2. Segmenty jsou nyní reprezentovány amplitudami a fázemi jednotlivých frekvencí. Amplitudy a fáze se upraví (např. převzorkováním, úpravou amplitud a fází vybraných frekvencí atd.)

3. Modifikované spektrum se převede zpět do časové oblasti pomocí inverzní STFT tak, že se každý segment převede inverzní FFT do časové oblasti a přičte se do výstupního signálu.

Změna délky signálu se provádí tak, že se při rekonstrukci zpětnou transformací do časové oblasti zvolí jiné časové měřítko.

Použití STFT není zcela triviální. STFT se používá v situaci, kdy potřebujeme zjistit změnu frekvence v čase. Chceme-li však zachytit velmi rychlé změny ve frekvenčním spektru, dostáváme se do potíží s rozlišením jednotlivých frekvencí. Čím vyšší časové rozlišení totiž požadujeme, tím kratší segmenty můžeme analyzovat pomocí FFT. Délka segmentu však přímo ovlivňuje rozlišení frekvencí. Z toho důvodu chceme-li lepší rozlišení frekvencí, musíme prodloužit segmenty, ale časové rozlišení se tím zhoršuje.

Fázový vokoder zlepšuje rozlišení frekvencí u krátkých segmentů pomocí výpočtu užívajícího dva různé segmenty. V jejich amplitudovém frekvenčním spektru nalezne maxima na shodné pozici a s pomocí jejich fázových rozdílů a rozdílu času mezi rámci výpočtem zpřesní frekvenční rozlišení [8].

Starší implementace fázového vokodéru vnášely do výstupního signálu různé zvukové artefakty, které byly způsobeny nepřesným zharmonizováním navazujících segmentů. Současné implementace tento problém částečně potlačily, avšak stále není zcela odstraněn.

Fázový vokodér může být použit i k jiným úpravám signálu než jen ke změně jeho délky. Je možné zvýšit základní tón, změnit jeho barvu atd.

Prokládání řečových rámců

Tento algoritmus jsem vytvořil pro první experimenty se změnou rychlosti řeči. Pracuje na úrovni řečových rámců. Jeho princip je velmi jednoduchý a spočívá ve využití jen některých rámců. Na příklad při využití každého 2. rámce získáme dvojnásobnou rychlost přehrávání. Rozhodování, který rámec se ve výstupním signálu uplatní a který nikoli, je řízeno čítačem rámců. Algoritmus využívá odlišné počítání při zvyšování rychlosti než při jejím snižování.

Zrychlení je určeno koeficientem změny rychlosti větším než 1. Žádný rámec se neopakuje, pouze se některé vynechávají. Paralelně se inkrementuje čítač rámců a „integruje“ konstantní funkce změny rychlosti. Při shodě celé část integrálu s počítadlem rámců se aktuální rámec použije, je-li různá, rámec se přeskočí. Tabulka 3.2 ilustruje postup.

čítač rámců	0	1	2	3	4	5
integrál	0,0	1,5		3,0	4,5	
výstupní signál	•	•		•	•	

Tabulka 3.1: Průběh čítače rámců a integrálu pro rychlost 1,5.

Zpomalení určuje koeficient v rozsahu $(0, 1)$. Při zpomalování řeči se žádné rámce nevynechávají, ale naopak se některé zopakují. Opět se využívá integrace funkce změny rychlosti. Pro každý rámec se provádí postupně integrace a dokud se nezmění celá část tohoto integrálu opakuje se aktuální rámec ve výstupním signálu.

integrál	0,0	0,4	0,8	1.2	1,6	2,0
použitý rámec	0	0	0	1	1	2

Tabulka 3.2: Průběh integrálu a použitých rámců pro rychlost 0,4.

Tento algoritmus je velmi jednoduchý a výpočetně skutečně nenáročný, avšak jeho nevýhody jsou značné. Velmi krátké hlásky, které se z vetší části mohou vejít do jednoho rámce můžou být při zvýšení rychlosti řeči snadno zcela vypuštěny a slovo, které je obsahovalo tak ztrácí svůj původní obsah. Hláska k má délku přibližně 8 ms, přičemž řečový rámec má délku 20 ms, k v takovém případě nepochybně zmizí. S větším zrychlením, kupříkladu trojnásobným jsou již zahozeny 2 následující rámce a hlásky vyslovené v tomto okamžiku jsou ztracené. I v případech, kdy je upravený signál srozumitelný, obsahuje řadu zvukových artefaktů způsobených napojováním řečových rámců bez přihlédnutí k fázi periody základního tónu v jaké se rámec nachází. Zvukové artefakty jsou způsobeny i tím, že se nijak neřeší plynulý přechod z jednoho rámce do druhého, ale pouze se k sobě připojují.

Kapitola 4

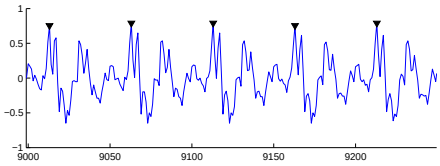
Metoda PSOLA

PSOLA (Pitch Synchronous OverLap Add) byla původně vyvinuta společností France Telecom.[1] Slouží k hladkému napojování hlasových vzorků, které již byly zaznamenány, přičemž umožňuje snadnou kontrolu nad základním tónem a délkou signálu[6, sekce 5.3.1]. Často se využívá u nástrojů na syntézu řeči, kdy se spojují předpřipravené části řeči. Takovéto syntetizéry se nazývají konkatenativní. Tyto syntetizéry dosahují vysoké kvality řeči při velmi nízké výpočetní složitosti – syntéza v reálném čase může byt provozována již na procesorech řady Intel 386 [1, str. 252]. Pitch Synchronous znamená, že segmenty hlasové nahrávky, které se spojují, jsou napojeny přesně tak, aby se překryly periody základního tónu. Tímto se zamezí zvukovým artefaktům ve výsledném signálu, jelikož spojované segmenty jsou „sfázovány“. PSOLA má několik variant. Varianta, která je použita v tomto projektu pracuje v časové oblasti, a je proto také někdy uváděna jako TD-PSOLA (Time Domain PSOLA). Ve své podstatě je velmi jednoduchá a výpočetně nenáročná. Dalšími variantami jsou FD-PSOLA (Frequency Domain) pracující ve frekvenční oblasti, LP-PSOLA (Linear Prediction) operující s lineární predikcí a dále WD-PSOLA, kde WD znamená Wavelet Domain – pracuje tedy s vlnkovou transformací [2].

Algoritmus PSOLA pracuje takto: V hlasovém signálu se určí pitch periody (periody základního tónu), přičemž přesnost jejich určení je velmi důležitá pro kvalitní zvukový výstup. Periody zpravidla začíná výrazným vrcholkem excitace. Tyto body se označují jako Pitch Marky (viz. obrázek 4.1). Spojování segmentů metodou překryj a sečti (OverLap Add) probíhá na úrovni pitch period a aby byl přechod mezi periodami co nejhladší, je nutné tyto periody vyseknout nějakou okénkovací funkcí. Často užívanou okénkovací funkcí je Hanningova funkce. V tomto projektu je okénkovací funkce zvolena poněkud jinak, k tomu se ale vrátím později v této kapitole. Okénkovací funkce se aplikuje v délce 2 až 4 period základního tónu platného v okolí pitch marku na jehož pozici je okénkovací funkce vystředěna. Při syntéze nového signálu se nejdříve vytvoří nová posloupnost (pozice) pitch marků a pitch marky v originálním signálu se na ně namapují. Metodou OLA se segmenty získané okénkovací funkcí „nasadí“ na pozice syntetizovaných pitch marků a sečtou se.

Manipulací se vzdáleností mezi syntetizovanými pitch marky můžeme docílit změnu základního tónu. Opakováním nebo vynecháním okénkovaných period základního tónu docílíme změnu délky syntetizovaného signálu, což je i náplň této práce.

[1] PSOLA/TD® je registrovaná obchodní značka společnosti France Telecom.



Obrázek 4.1: Pitch marky

4.1 Detekce základního tónu

Detekce základního tónu a s ním spojené určení pozic pitch marků je kritické pro kvalitní zvukový výstup. Pokud se vytváří fonémová databáze pro systémy syntézy řeči je možné použít manuální určení, které má nejvyšší přesnost, avšak je to proces velmi zdlouhavý. Pro určení základního tónu se tak využívá několik algoritmů pracujících automaticky. Jejich přehledem začnu následující část dokumentu.

4.1.1 Autokorelační funkce

Nejjednodušším algoritmem pro detekci základního tónu je autokorelační funkce (CCF – Cross-Correlation Function). Tento algoritmus pracuje v časové oblasti nad jedním rámcem. Princip spočívá v nalezení posuvu rámce vůči sobě sama, při kterém si je signál nejpodobnější. Podobnost se zjišťuje při posunutém signálu vynásobením odpovídajících vzorků, přičemž se tyto součiny sčítají. Čím větší je shoda signálu s jeho posunutým obrazem, tím vyšší je tato suma. Ohodnocení podobnosti se provede pro všechna posunutí a maximum z nich nám udá posunutí, jehož hodnota odpovídá periodě základního tónu. Nejvyšší hodnotu má autokorelační funkce s nulovým posunutím, tento nulový koeficient ale nezapočítáváme. V praxi se posouvá signál v rozmezí hodnot odpovídajících základnímu tónu hlasu. Např. při hledání základního tónu mužského hlasu 80 Hz – 160 Hz se bude výpočet provádět pro posun od 50 do 100 vzorků při vzorkovacím kmitočtu 8000 Hz. Matematicky je autokorelační funkce vyjádřena vztahem 4.1.

$$R(j) = \sum_{n=1}^{N-j} (x[n]x[n-j]) \tag{4.1}$$

Autokorelační funkce pracuje dobře na rámcích se znělým hlasovým signálem, avšak v neznělých oblastech je značně nestabilní. Je tedy nutné určit znělé oblasti a v místech neznělých základní tón vhodným způsobem aproximovat. Dalším omezením autokorelace je vzorkovací frekvence. Jelikož pracuje přímo se vzorky a jejich posunutím, má při nízké vzorkovací frekvenci nízké rozlišení frekvence základního tónu.

AMDF (Average Magnitude Difference Function)

Jedná se o obdobu funkce CCF avšak výpočetně méně náročnou. Místo součinu používá rozdíl a namísto hledání maxima hledá minimum. Matematické vyjádření AMDF je v rovnici 4.2.

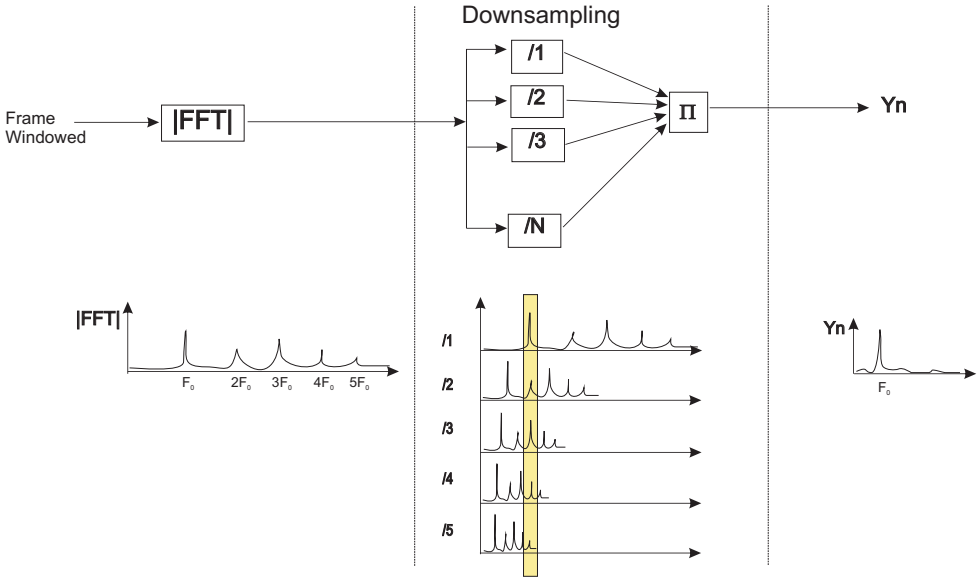
$$R(j) = \sum_{n=1}^{N-j} |x[n] - x[n-j]| \tag{4.2}$$

S autokorelační funkcí jsem v počátku experimentoval, ovšem výsledky byly neuspokojivé. Určený základní tón byl často zcela špatný. Problémové bylo též určení znělosti daného rámce, které by stanovilo, ve kterých oblastech signálu jsou hodnoty základního tónu zjištěného CCF platné a ve kterých oblastech jsou naopak tyto hodnoty neplatné a základní tón by se zde vhodným způsobem interpoloval.

4.1.2 HPS (Harmonic Product Spectrum)

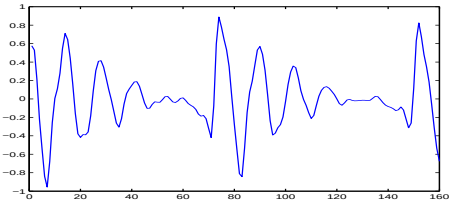
Tato metoda pracuje ve frekvenční oblasti. Vychází z předpokladu, že řečový signál je výsledkem filtrování základního tónu v artikulačním ústrojí. Při tomto filtrování se výstupní signál skládá převážně ze samotného základního tónu a harmonických složek, které jsou jeho celočíselným násobkem. Algoritmus tedy převede zkoumaný řečový rámec do frekvenční oblasti pomocí rychlé Fourierovy transformace a vzniklé spektrum následně několikrát podvzorkuje. Při podvzorkování frekvenčního spektra na polovinu, třetinu, čtvrtinu atd. se harmonické složky zarovnávají na pozici základního tónu. Při vynásobení průběhů několika podvzorkování vystoupí jedno maximu, které bude na pozici základního tónu. Výhodou je relativně nízká citlivost na šum a výpočetní nenáročnost. Nevýhodou je však nepřesnost na nízkých frekvencích. Pro kvalitní výsledky je potřeba mít delší řadu FFT [7]. Funkci HPS nejlépe ilustruje obrázek 4.2.

Jelikož HPS pracuje s násobky základního tónu, je zřejmé, že hodnoty zjištěné při vyslovování neznělých fonémů nebudou platné. Pro překlenutí neznělých oblastí jsem použil dynamického programování a interpolace, tedy postup uplatňovaný v Trasovači spojitého základního tónu z kapitoly 4.1.3.

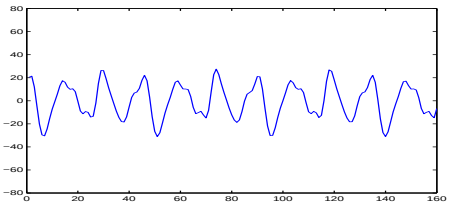


Obrázek 4.2: Harmonic Product Spectrum

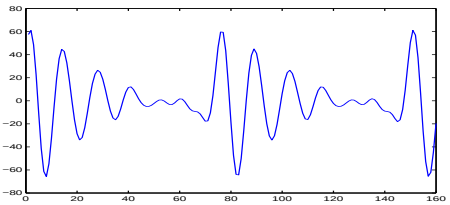
Algoritmus HPS je v projektu implementován. Jeho výstupy jsou ovšem nestálé. Některé zvukové ukázky určil poměrně přesně, jiné byly zcela nepoužitelné. Testováním se potvrdil neduh zmiňovaný v předchozím odstavci – malá přesnost na nízkých frekvencích. Nahrávky s ženským hlasem, který má vyšší základní tón, byly často určeny přesně. Naopak nahrávky mužského hlasu s nižším základním tónem byly správně určeny zřídkakdy.



Originální signál



$F_0 = 80Hz$



$F_0 = 108Hz$

Tabulka 4.1: Originální signál a dvě jeho rekonstruované podoby – se špatným základním tónem 80 Hz a se správným 108 Hz. Pro rekonstrukci bylo použito 10 harmonických složek. Je jasně vidět, že při správném základním tónu se generovaný signál velmi podobá originálu.

4.1.3 Trasovač spojitého základního tónu na základě modelu harmonických a šumu (HNM)

Tento algoritmus byl vyvinut na naší fakultě[10]. Je velmi sofistikovaný a podává vynikající výsledky. Výhodou je, že není výrazně citlivý na zašumělé signály. Podobně i změna vzorkovací frekvence signálu nevyžaduje jeho úpravy. Ze své podstaty umí doplnit pitch marky i v neznělých oblastech, což je konkrétně v mém projektu velmi výhodná vlastnost.

Stejně jako jiné algoritmy i tento pracuje s rámci. První část algoritmu analyzuje jednotlivé rámce a druhá využívá výsledků této analýzy a určí sled pitch marků v průběhu celého signálu pomocí dynamického programování.

Podobně jako u algoritmu HPS, vycházíme z předpokladu že signál je složen především z harmonických složek základního tónu. Známe-li tedy základní tón a jeho harmonické včetně jejich modulů a fází můžeme zrekonstruovat signál, který bude velmi podobný původnímu. Základní tón je však to, co zde hledáme, a tak zkusíme zrekonstruovat signál pro všechny základní tóny v přípustném rozsahu – např. 50 Hz – 350 Hz a najít takový základní tón, který generuje signál s nejmenší chybou. Chybu definujeme jako rozdíl signálu skutečného a rekonstruovaného $E(t) = O(t) - S(t)$.

Informace o modulech a fázích frekvenčních složek signálu získáme pomocí rychlé Fourierorovy transformace. V tomto okamžiku můžeme začít rekonstruovat signál pro jednotlivé základní tóny. Pro každý základní tón využijeme prvních 5–10 násobků základního tónu (harmonických) a s jejich pomocí syntetizujeme nový signál. Postup popisuje následující rovnice 4.3 a výsledky takovéto rekonstrukce jsou zobrazeny v tabulce 4.1

$$S_{F_0}(t) = \sum_{i=1}^{n} A_i cos(iF_0 + \varphi_i), \tag{4.3}$$

kde n je počet harmonických a amplitudy A_i a fáze φ_i pochází z FFT. Nový signál může mít odlišnou energii, což způsobí větší chybové hodnoty než by skutečně měly být. Z

tohoto důvodu se snažíme chybu minimalizovat výpočtem zisku:

$$G_{F_0} = \frac{\sum_t O(t) S_{F_0}(t)}{\sum_t S^2_{F_0}(t)} \tag{4.4}$$

Chybu po korekci ziskem pak vypočítáme podle vztahu:

$$E_{F_0}(t) = \sum_t O(t) - S_{F_0}(t) G_{F_0} \tag{4.5}$$

Čím více harmonických se použije při rekonstrukci signálu, tím lépe se obraz se svoji předlohou schoduje při správném základním tónu. Vyneseme-li na graf průběh chyb v závislosti na základním tónu, budou patrná minima u skutečného základního tónu a jeho harmonických složek. Někdy se může stát, že chyba na pozici nějaké harmonické složky je menší než chyba skutečného základního tónu. Tomuto můžeme zabránit budeme-li postupně sčítat chybové hodnoty u signálu generovaného jednou, dvěma a postupně až desíti harmonickými. Postup vyjadřuje rovnice 4.6.

$$E^s_{F_0}(t) = \sum_{n=1}^{h} E^n_{F_0}(t), \tag{4.6}$$

kde h je celkový počet využívaných harmonických složek a $E^n_{F_0}(t)$ je chyba pro rekonstruovaný signál pomocí n harmonických.

Tímto krokem byla dokončena první část algoritmu, kdy jsme pracovali s jedním rámcem. V tomto okamžiku by bylo možné určit základní tón pro znělé oblasti prostým vyhledáním minimální chyby, neznělé oblasti by ale nadále byly problematické. Neznělé hlásky - zejména frikativy (f, s, š, ch), které mají charakter šumu, nejsou ovlivněny základním tónem mluvčího a při výpočtu chyby, tak jak byl uveden v předchozím odstavci, by se nenalézala žádná relevantní mimina. Druhá část algoritmu však využívá nabyté informace k určení základního tónu v průběhu celého signálu.

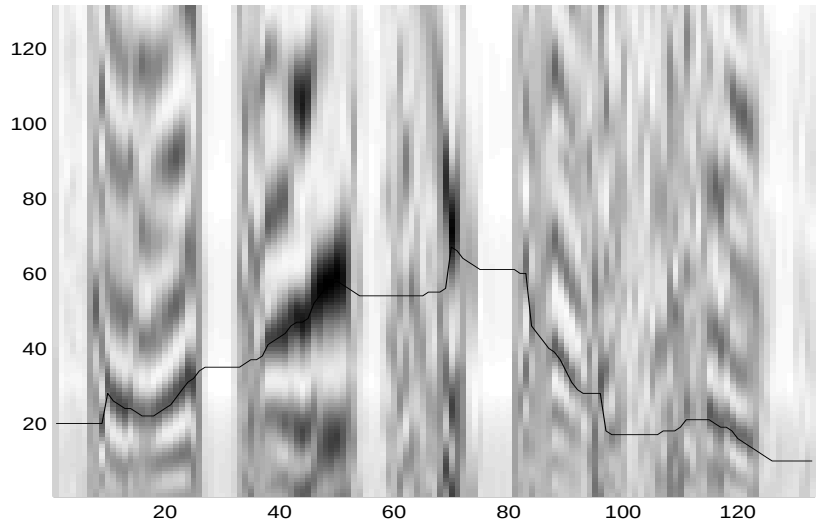
Určení základního tónu v rámci celého signálu probíhá nad maticí chyb, kde vodorovná osa vyjadřuje čas v rámcích a svislá osa pak základní tón. Grafické znázornění této matice je na obrázku 4.3.

Z obrázku jsou jasně patrné světlejší svislé pruhy – to jsou oblasti, kde je buď pauza, nebo neznělé hlásky. Naopak v oblastech se znělým signálem je jasně patrná tmavá „vlna“ a několik slabších vlnek, což jsou chyby u správného základního tónu a jeho harmonických složek. V grafu je již zobrazen průběh základního tónu, který je vypočítán pomocí následujícího postupu.

Dynamické programování

Jak již bylo zmíněno dříve, v neznělých oblastech nebo dokonce v místech ticha není možné správně přímo určit základní tón, protože tam jednoduše žádný není. Jelikož ho tam ale potřebujeme, resp. potřebujeme pitch marky, musíme jej nějakým způsobem doplnit. Efektivním řešením tohoto problému je Dynamické programování.

Dynamické programování pro nalezení cesty skrze nějakou matici má několik variant. Pro náš problém, kdy hledáme nejkratší cestu s nejnižší cenou (chybou) je nejvýhodnější algoritmus, který je označován jako „problém obchodního cestujícího“. Obchodní cestující



Obrázek 4.3: Matice chyb základního tónu v rámcích. Základní tón začíná na ose Y v hodnotě 0, ale ve skutečnosti je to 70, jelikož graf ukazuje rozsah základního tónu od 70 do 200 Hz. Odstíny šedé barvy vyjadřují hodnotu chyby, čím tmavší, tím menší chyba. Černá linka procházející napříč grafem zobrazuje výsledný základní tón v rámcích.

hledá nejkratší cestu z jednoho konce kontinentu na druhý (z levé strany matice na pravou) a po cestě prochází města, ve kterých musí přespat, za což musí zaplatit (hodnota chyby v matici). Obchodník se snaží minimalizovat cenu za přepravu mezi městy a zároveň započítává cenu noclehu, takže někdy vyhledá třeba i delší cestu, na jejímž konci je výrazně levnější nocleh, než kdyby se vydal krátkou cestou. Postup jakým je nalezení nejkratší cesty realizováno již dále popíši na naší matici chyb.

Hledání probíhá ve dvou průchodech – dopředném a zpětném.

Dopředný průchod (forward pass)

V tomto průchodu procházíme v každém rámci každý základní tón a zjistíme z jakého základního tónu předchozího rámce jsme se do něj mohli dostat. Rozsah základních tónů předchozího rámce, které analyzujeme, je omezen maximální změnou základního tónu mezi rámci ΔF_0 – například 20 Hz. Do základního tónu $F_0(t)$ tedy můžeme vstoupit ze základního tónu předchozího rámce $F'_0(t-1) \in \langle F_0(t) - \Delta F_0, F_0(t) + \Delta F_0) \rangle$. Ze zkoumaných základních tónů předchozího rámce si zapamatujeme ten, pro který je cena přechodu do aktuálního základního tónu nejmenší. Cena přechodu se určí výpočtem 4.7

$$W_{F_0}(t) = W_{F'_0}(t-1) + E^s_{F_0}(t) + C\sqrt{(F'_0(t-1) - F_0(t))^2 + 1}\ \frac{E^s_{F'_0}(t-1) + E^s_{F_0}(t)}{2}, \quad (4.7)$$

kde:

$W_{F_0}(t)$ je cena do aktuálního základního tónu.

$W_{F_0'}(t-1)$ je cena cesty do základního tónu předchozího rámce.

$E^s_{F_0}(t)$ je chyba aktuálního základního tónu.

C je váhový koeficient.

$\sqrt{(F_0'(t-1) - F_0(t))^2 + 1}$ je délka cesty ze základního tónu předchozího rámce do aktuálního základního tónu.

$\frac{E^s_{F_0'}(t-1) + E^s_{F_0}(t)}{2}$ je průměrná chyba základního tónu aktuálního a předchozího.

Koeficient C nastavuje preferenci mezi nejkratší cestou a cestou s nejmenší chybou.

Zpětný průchod (backward pass)

Zpětný průchod je velmi jednoduchý a rychlý. V posledním rámci najdeme základní tón s nejmenší cenou, kterou jsme zjistili v prvním průchodu. Ke každému základnímu tónu jsme si zapamatovali nejvýhodnější základní tón předchozího rámce, takže nyní pouze stačí projít zpětně základní tóny, přičemž u každého ihned víme jeho předchůdce a do něj také vkročíme až se postupně dostaneme na začátek matice.

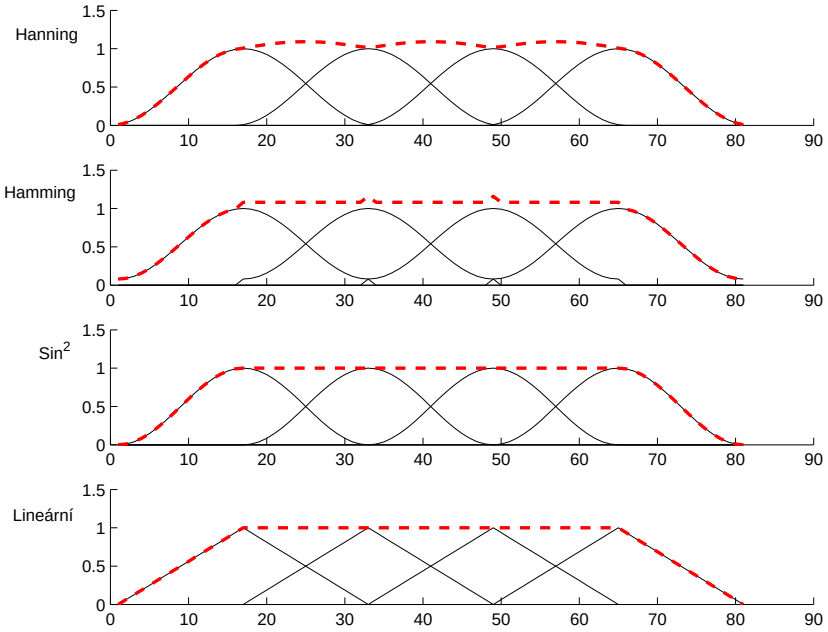
Určení Pitch Marků

Nyní je znám základní tón v jednotlivých rámcích a následuje určení poloh pitch marků, neboli začátků period základního tónu. K tomuto účelu je nezbytné určit, které rámce jsou znělé a které neznělé. Tento úkol se nejlépe řeší pomocí chyby na trase základního tónu, která byla určena dynamickým programováním. Chyba se podél trasy základního tónu mění a jelikož je normovaná do rozsahu $\langle 0, 1\rangle$ je možné použít metodu prahování – od jisté hodnoty chyby se rámec považuje za neznělý. Naopak pod touto hodnotou je rámec znělý.

Samotné určení pitch marků probíhá následovně. Jako výchozí bod se zvolí první vzorek signálu. Rámec v němž se tento vzorek nachází teď nese informaci o základním tónu a znělosti. Je-li rámec znělý, pak se použije jeho základní tón a jeho perioda se připočte k onomu prvnímu vzorku. Nyní se v okolí vypočítané hodnoty najde maximum signálu – to by mělo připadnout na vrcholek excitace periody základního tónu. Takto byla určena pozice dalšího pitch marku a můžeme začít hledat další. Skočí-li se do neznělé oblasti, odpadá hledání lokálního maxima v místě, kam ukáže součet aktuálního vzorku a periody základního tónu. Výsledek součtu se vezme jako další pitch mark a celá procedura „skoku“ začne znovu.

4.2 Skládání period základního tónu

V úvodu kapitoly jsem hovořil o nutnosti okénkování period základního tónu pro jejich následné spojování. Běžně používanými okénkovacími funkcemi jsou Hanningova a Hammingova. Pro případ navazování segmentů metodou OLA ovšem nepříznivě ovlivňují výstupní signál. Naším cílem je pouze změnit rychlost řečového signálu, jmenované funkce ale zvýší i jeho energii. Součet okénkovací funkce dvou sousedních segmentů by tedy měl nabývat hodnoty 1. Jako okénkovací funkci jsem zvolil funkci sin^2. Funkce sin má tu vlastnost, že při posunutí o čtvrt periody má stejný průběh jako funkce cos. Jelikož segment se okénkuje přes dvě periody základního tónu a při navazování se přes sebe periody překrývají, dojde tak vlastně k překrytí funkcí sin^2 a cos^2. Součtem těchto funkcí je 1. Tvar takovéto



Obrázek 4.4: Okénkovací funkce. Okénkovací funkce jsou vykresleny tenkou spojitou čarou, jejich součet pak čárkovanou tlustou.

okénkové funkce rovněž preferuje část kolem středu pitch marku a mírní tak možné interference mezi sousedními segmenty. Vlastnost součtu rovného 1 má i lineární okénkovací funkce, ovšem její průběh způsobuje větší promísení sousedních segmentů. Průběhy jmenovaných okénkovacích funkcí jsou na obrázku 4.4.

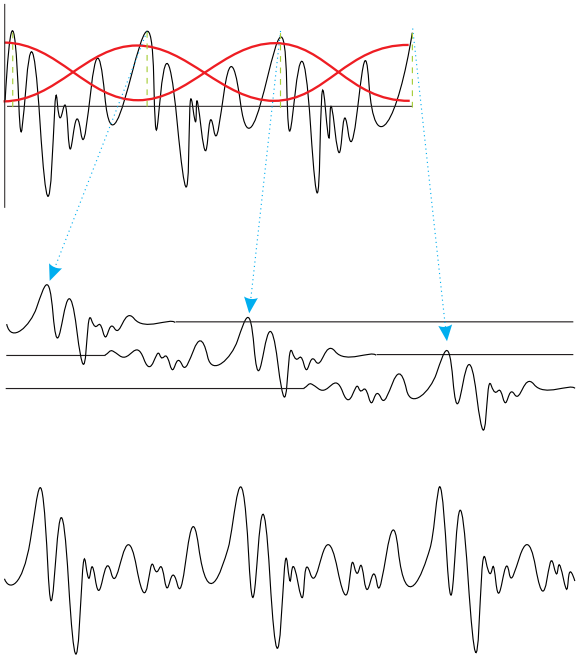
K popisu skládání period základního tónu použiji obrázek 4.5. Horní část obrázku znázorňuje originální signál respektive jeho znělý segment, ve kterém je snadno rozpoznatelná jeho periodičnost. Na tomto obrázku jsou čtyři pitch marky, které jsou naznačeny svislými čárkovanými čarami. Všechny označují vrchol excitace (buzení hlasového traktu).

Okénkovací funkce vybírá 2 periody základního tónu, které se následně překrývají s dalšími segmenty. Každá perioda základního tónu je vlastně součástí dvou okénkovaných segmentů. V průběhu signálu se délka periody základního tónu mění a v závislosti na ní se mění i délka okénkovací funkce. V ilustraci PSOLy jsou jednotlivé segmenty vyseknuté okénkovací funkcí zobrazeny ve střední části.

Podle záměru užití metody PSOLA volíme nové pozice okénkovaných segmentů v syntetizovaném signálu. Hovoříme o vytvoření syntetizovaných pitch marků. Jejich tvorbě se budu věnovat v sekci Výběr period základního tónu 4.3. Známe-li pozice segmentů je poslední operací jejich sečtení tak, jak je ilustrováno na dolní části obrázku 4.5.

4.3 Výběr period základního tónu

Až doposud byl popis metody PSOLA shodný jak pro variantu změny základního tónu, tak pro variantu změny délky signálu. To, co tyto dva režimy rozlišuje, je výběr period



Obrázek 4.5: Funkce metody PSOLA při snížení základního tónu

základního tónu, které mají být použity v syntetizovaném signálu a jejich umístění v rámci tohoto signálu.

Pro variantu změny základního tónu měníme vzdálenost mezi pitch marky. Například na obrázku 4.5, na kterém jsem ilustroval funkci PSOLy, dochází ke snížení základního tónu. Původní signál měl periodu základního tónu danou vzdálenosti mezi jednotlivými pitch marky. Ve výstupním signálu však byly tyto vzdálenosti, a tedy periody základního tónu, prodlouženy, čímž poklesl základní tón. Naopak pokud by byly segmenty syntetizovány do výstupního signálu s užšími rozestupy, došlo by ke zvýšení základního tónu. V zájmu zachování délky signálu může být nutné některé periody základního tónu zopakovat (při zvyšování základního tónu) anebo vypustit (při snížení základního tónu).

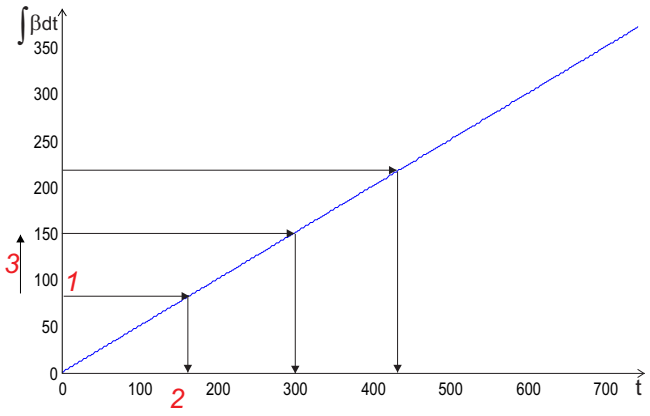
Tento projekt řeší změnu rychlosti řeči, což je varianta změny délky signálu. V tomto případě se opakují nebo vynechávají jednotlivé periody základního tónu, přičemž si zachovávají vzdálenosti mezi sousedními pitch marky. Ve skutečnosti může dojít k mírným změnám vzdálenosti mezi pitch marky v situaci, kdy se vypustí nějaká perioda, a periody, které se na sebe budou přikládat mají různou délku. Tato nerovnost se řeší tak, že se zvolí délka navazované periody (tedy té „pozdější“).

Nyní k samotnému výběru period, potažmo pitch marků, které v syntetizovaném signálu budou použity a ty které v něm použity nebudou. Vstupním parametrem pro toto rozhodování je poměr délky nového signálu k délce stávající nazvaný β. Hodnoty $\beta \in (0, 1)$ znamenají zkrácení délky signálu, což se projeví jako rychlejší tempo řeči a hodnoty větší než 1 signál prodlouží, a posluchači se bude řeč jevit jako pomalejší.

K určení period, které mají být vynechány a které zopakovány pomůže integrace funkce změny délky signálu 4.8. V našem případě je funkce konstantní β. Integrací konstantní funkce získáme přímku, která bude mít směrnici rovnu změně délky signálu. Graf tohoto integrálu je na obrázku 4.6. Na ose x je vynášen čas ve vzorcích, osa y reprezentuje integrál

změny délky signálu.

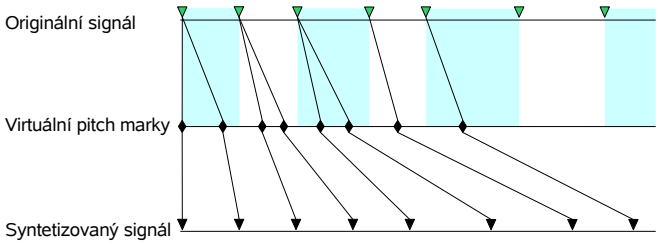
$$y = \int \beta dt \tag{4.8}$$



Obrázek 4.6: Určení syntetizovaných pitch marků pro změnu délky signálu 0,5

Syntetizované a virtuální pitch marky

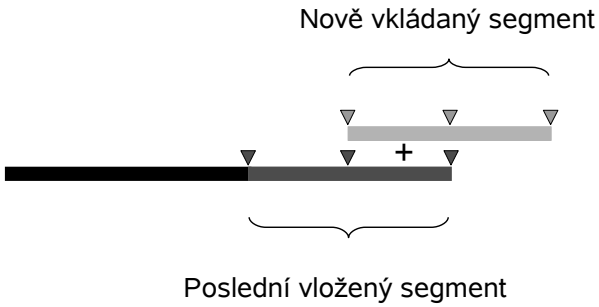
Na obrázku 4.6 je znázorněn postup při vytváření tzv. syntetizovaných pitch marků. Syntetizované pitch marky umožní výběr právě těch period základního tónu, které se promítnou do výsledného signálu. Vyhledávání začíná tím, že první syntetizovaný pitch mark nabude hodnoty prvního skutečného pitch marku. Další postup je iterativní – v prvním kroku se na grafu integrálu vyhledá hodnota syntetizovaného pitch marku (1). Vzorek z osy x (krok 2), který tuto hodnotu nesl (nazvěme jej virtuální pitch mark), spadá do nějaké periody základního tónu a její délka se nyní připočte k původně vyhledávanému syntetizovanému pitch marku (krok 3), čímž se získá nový syntetizovaný pitch mark. Kroky 1 až 3 se opakují až do dosažení konce signálu. Ke každému syntetizovanému pitch marku, který leží na ose y, se tak mapuje i virtuální pitch mark ležící v ose x. Toto mapování je zobrazeno v obrázku 4.7



Obrázek 4.7: Mapování syntetizovaných pitch marků na virtuální.

Virtuální pitch marky jsou v časové línii originálního signálu a určují, které periody základního tónu se použijí při syntéze nového signálu. Virtuální pitch mark spadá svojí pozicí do nějaké periody základního tónu, která začíná svým skutečným pitch markem.

Pitch mark, který jej bezprostředně předchází a další, který jej následuje, určují hranice segmentu, který se vybere okénkovací funkcí. Nový segment se naváže do syntetizovaného signálu tak, že jeho levý kraj se zarovná na středový pitch mark posledního segmentu. Postup sčítání je na následujícím obrázku.



Obrázek 4.8: Operace OLA na segmentech, které jsou zarovnané na pitch marky. Trojúhelníky značí pitch marky.

Kapitola 5

Rozšíření o fonémový rozpoznávač

Do tohoto okamžiku byla popisována metoda PSOLA, která pracovala s řečovým signálem aniž by rozlišovala, jaký je obsah řeči – jaké hlásky se vyslovují. Výsledkem takového přístupu je, že i hlásky jako *a* a *p* doznaly stejné změny délky. Charakter těchto dvou hlásek je však velmi odlišný, a z tohoto důvodu si zaslouží poněkud propracovanější přístup. Stručným výkladem o hláskách – fonémech otevřu tuto kapitolu.

5.1 Fonémy

Foném je zvukový jazykový prostředek sloužící k odlišení morfémů, slov a tvarů slov téhož jazyka s různým významem. Liší se od ostatních fonémů téhož jazyka nejméně jednou fonologickou distinktivní vlastností (např. kvantita – délka trvání fonému). Současná spisovná čeština rozlišuje 36 fonémů [5]. Ovšem soubor používaných fonémů je velmi živý a i někteří autoři se na sestavě fonémů neshodují. Například diftong *ou* někteří zařazují jako foném, jiní jej chápou jako dva fonémy *o* a *u*. Aplikace, jež byla vyvinuta v rámci diplomového projektu, převzala českou fonémovou sadou z knihovny BSAPI. Tato sada je definována v tabulce A.1 a obsahuje i některé zástupné fonémy, které představují různé chyby v signálů anebo ruchy na pozadí.

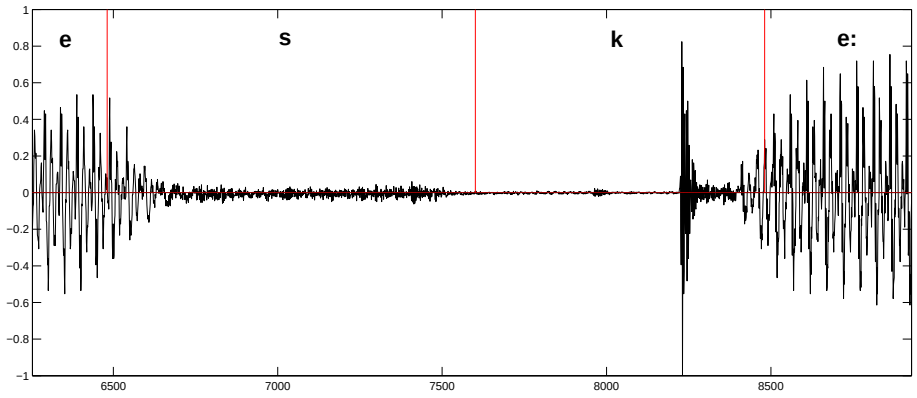
Česká fonologie rozlišuje dvě hlavní skupiny hlásek – vokály a konsonanty. V následujících odstavcích budou blíže popsány.

5.1.1 Vokály – samohlásky

Vokály jsou takové hlásky, jejichž charakteristickým rysem je tón. Na rozdíl od souhlásek nevzniká při jejich artikulaci šum. Dominantní vliv na vyslovovanou samohlásku má poloha jazyku a tvar rtů. Zvukový signál všech samohlásek je nápadně periodický, což je dáno tím, že vokály jsou tvořeny pouze základním tónem z hlasivek, který je formován rezonancí v rezonančních dutinách – především ústní. Čeština má 10 fonémů spadajících do skupiny vokálů: *a*, *e*, *i*, *o*, *u*, *a:*, *e:*, *i:*, *o:*, *u:*. Jiné jazyky k vokálům řadí např. i foném *j*. Grafická ukázka vokálu *e:* je na obrázku 5.1.

5.1.2 Konsonanty – souhlásky

U konsonantů dochází k vytvoření překážky v cestě výdechovému proudu – *striktury*. Tvoření striktury má několik fází: *intenzi* (vytvoření přehrady) – *tenzi* (podržení překážky) – *detenzi* (návrat do klidového postavení nebo přechod k další hlásce). Tyto fáze jsou dobře



Obrázek 5.1: Úsek ze slova „české“ s označením hranic fonémů tak, jak je detekoval fonémový rozpoznávač.

pozorovatelné u tzv. úžinových souhlásek, kdežto u závěrových, u nichž výslovnost nastává až v okamžiku zrušení překážky, mluvíme o *exkurzi* a *rekurzi* nebo o *implozi* a *explozi*.

Podle typu striktury rozlišujeme několik typů konsonantů:

- **Závěrové** (okluzní) souhlásky vznikají při úplném přerušení výdechového proudu a jeho následném uvolnění. Patří sem například *d*, *t*, *n*.
- **Úžinové** (konstriktivní) hlásky, při nichž výdechový proud prochází po celou dobu výslovnosti zúženým místem, jsou např. *s*, *z*, *f*.
- **Polozávěrové** (semiokluzní) vznikají kombinací závěrových a úžinových souhlásek. Na počátku se vytváří slabý závěr, který se v průběhu vyslovování jediné hlásky mění v úžinu. Polozávěry jsou *c* a *č*.
- **Kmity** (vibrace) U kmitů se úžina při vyslovování jediné hlásky mění – zvětšuje a zmenšuje. Vibranty jsou *r* a *ř*.

Na obrázku 5.1 jsou zaznamenány průběhy dvou konsonantů – úžinové *s* a závěrové *k*. Tabulka 5.1 přehledně zobrazuje rozdělení souhlásek podle místa a způsobu tvoření v artikulačním ústrojí.

5.1.3 Flexibilita fonémů

Z pohledu této práce nazývám flexibilitou fonému to, jak lehce je možné změnit jejich délku bez výraznějšího vlivu na jejich zvukovou kvalitu.

Vokály

Vokály mají periodický průběh. Periodické děje se dají poměrně lehce „nastavovat“ tak, že se na sebe periody opakovaně navazují. Důkazem toho je i fakt, že vokály mají pět fonému krátkých a pět dalších fonémů, které se od prvních pěti liší pouze v kvantitativní vlastnosti – tedy jsou různě dlouhé. Napr. *a* – *a:*.

Podle způsobu tvoření		Podle místa tvoření													
		Retné				Dásňové				Tvrdo-patrové		Měkko-patrové		Hrtanové	
		Retoretné		Retozubné		Přední		Zadní							
znělost		n	z	n	z	n	z	n	z	n	z	n	z	n	z
Závěrové	Ústní	p	b			t	d			ť	ď	k	g		
	Nosní				μ		n				ň		ŋ		
Polozávěrové						c	dz	č	dž						
Úžinové	Středové		u	f	v	s	z	š	ž		j	x			h
	Bokové						l								
	Kmitavé						r,ř								

Tabulka 5.1: Přehled českých konsonantů

Konsonanty

Konsonanty periodický průběh zpravidla nemají. Část z nich, která má charakter šumu – např. *s*, *f* je poměrně flexibilní. Šum, ač neperiodický, se dá také navazováním šumových segmentů prodlužovat. Česká sada fonémů sice nemá krátké a dlouhé *s*, fyzicky jej ale není problém vyslovit.

Zcela jiným případem jsou ale ostatní konsonanty, tedy ty, které nejsou ani periodické, ani šumové. Vezměme si například souhlásku *k*, která je i na obrázku 5.1. Jedná se o závěrovou souhlásku, tedy foném tvořící se v krátkém okamžiku při odstranění překážky – v tomto případě jazyka opírajícího se svým kořenem o měkké patro. Na obrázku fonému *k* je těsně před vyslovením tohoto fonému prakticky nulový signál, při uvolnění překážky dojde k „explozi“, která během přibližně 8 ms přejde do téměř klidového stavu a počátku fonému *e:*. Je zřejmé, že takovýto signál nemá velkou flexibilitu a měnit jeho délku je možné jen na úkor srozumitelnosti.

5.2 Rozšíření PSOLy

PSOLA sama o sobě „neví“ jaké fonémy právě zpracovává. Nyní, když víme, že tato informace má pro změnu délky řečového signálu hodnotu, je potřeba ji nějakým způsobem zjistit a PSOLe zprostředkovat. K tomuto účelu bude zapotřebí fonémového rozpoznávače a modifikace části PSOLy.

5.2.1 Fonémový rozpoznávač

Fonémový rozpoznávač detekuje fonémy v řečovém signálu. Do tohoto diplomového projektu byl jako fonémový rozpoznávač použit modul z knihovny BSAPI vyvíjené na naší fakultě. Fonémový rozpoznávač pracuje s fonémovou sadou z tabulky A.1. Fonémový rozpoznávač je založený na hierarchických neuronových sítích. Verze, která byla při vývoji dostupná pracuje pouze na vzorkovacím kmitočtu 8000 Hz, čímž je limitováno nasazení při zpracování vstupních signálů. Pro využití této knihovny pro zpracování signálů s vyšší vzorkovací frekvencí jsou dvě řešení. Prvním řešením by bylo natrénovat neuronovou síť BSAPI na vzorcích s vyšším kmitočtem, což by ovšem opět limitovalo BSAPI pouze na

naučený vzorkovací kmitočet. Druhým řešením je signál před jeho zpracováním v BSAPI za běhu převzorkovat na frekvenci 8000 Hz. Tuto možnost jsem netestoval.

Do tohoto rozpoznávače vstupuje řečový signál a na výstupu jsou následně dostupné informace o nalezených fonémech. Konkrétně je možné zjistit název fonému, v jakém časovém úseku <od, do> se foném nalézal a s jakou jistotou byl tento foném určen. Pro nasazení v této diplomové práci je užitečný pouze název fonému a kdy začíná a končí. Prakticky se ovšem z rozpoznávače využívá jen název fonému a kdy končí, jelikož signál je celý pokryt fonémy a konec jednoho je tedy i začátkem druhého.

5.2.2 Modifikace výběru period základního tónu

Klíčem ke změně délky signálu pomocí metody PSOLA je opakování, či naopak vynechávání některých period základních tónů. K určení period, které se do výstupního signálu promítnou a které ne, sloužil postup popsaný v sekci 4.3. A právě tohoto postupu se bude týkat následující úprava.

Jak bylo ve zmíněné sekci uvedeno, délka signálu je určena koeficientem β. Pro výběr period základního tónu se nejdříve tento koeficient integroval a z výsledného grafu se určovalo, která perioda a kolikrát se použije. Jelikož algoritmus nerozlišoval mezi fonémy, byl koeficient β konstantní a výsledný integrál měl podobu přímky. K tomu, aby byla PSOLA *phoneme-aware*, je nutné změnit koeficient β tak, aby nebyl konstantní, ale měnil se v čase podle fonémů v řečovém signálu.

Zapojení flexibility

Výpočet pro proměnnou β – nazvěme ji β', bude využívat původní koeficient, který nyní určuje očekávanou změnu délky signálu, a dále flexibilitu fonému vyjádřenou v rozmezí <0, 1>. Nazvěme flexibilitu γ. Flexibilita s hodnotou 0,0 znamená, že foném se nedá upravovat. Hodnota 1,0 značí libovolně pružný foném. Hodnoty mezi těmito dvěma extrémy vyjadřují poměrnou část z očekávané změny délky, kterou daný foném akceptuje. Např.: Má-li foném flexibilitu 0,5, pak při požadavku na změnu délky danou $\beta = 1{,}5$, bude mít výsledná upravená β' hodnotu 1,25, jelikož původní požadavek byl prodloužit signál o polovinu, avšak foném toleruje pouze půlku z požadované změny.

Výpočet upravené β je dán vztahem 5.1.

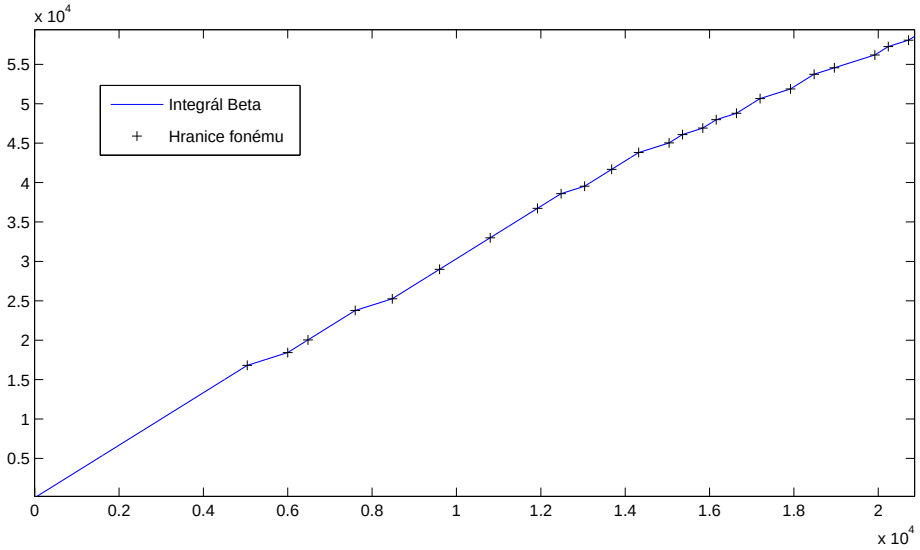
$$\beta' = 1.0 + (\beta - 1.0)\gamma; \tag{5.1}$$

Výsledkem této úpravy je nový vztah pro integrál β:

$$y = \int \beta'(\beta, \gamma)dt \tag{5.2}$$

Výsledný graf upravené výběrové funkce je na obrázku 5.2. Tento integrál již nemá podobu přímky, ale zvlněné křivky. Křížky na této křivce označují rozhraní sousedních fonémů.

V projektu byly pro pro jednotlivé fonémy nastaveny hodnoty γ z tabulky A.2. Všechny vokály, ke který se v BSAPI řadí i *au, eu, ou*, mají plnou flexibilitu, tedy 1,0. Poslechem a zkoumáním grafických průběhů fonémů jsem dospěl k rozdělení konsonantů do dvou až třech skupin z pohledu jejich flexibility. Plnou flexibilitu mají jen souhlásky úžinové. Polozávěrové hlásky tvoří přechod k méně flexibilním fonémům a na základě poslechových experimentů jim byly přiřazeny hodnoty 0,7 nebo 0,3. Nakonec Závěrové souhlásky mají v současném



Obrázek 5.2: Modifikovaný integrál, při změně rychlosti 0,3.

ohodnocení nejmenší flexibilitu 0,3. Jelikož je rozdělení poměrně hrubé, nabízí se možnost natrénovat fonémový rozpoznávač tak, aby rozpoznával jen tyto tři skupiny fonémů – tedy vokály, úžinové souhlásky a závěrové souhlásky a případně i polozávěrové. Aktuální verze fonémového rozpoznávače podává při jeho současné sadě fonémů dobré výsledky, avšak při zjednodušení dělení fonémů by se mohlo dosáhnout ještě větší přesnosti v rámci nového dělení. Jednodušší dělení by stále plně postačovalo požadavkům tohoto projektu.

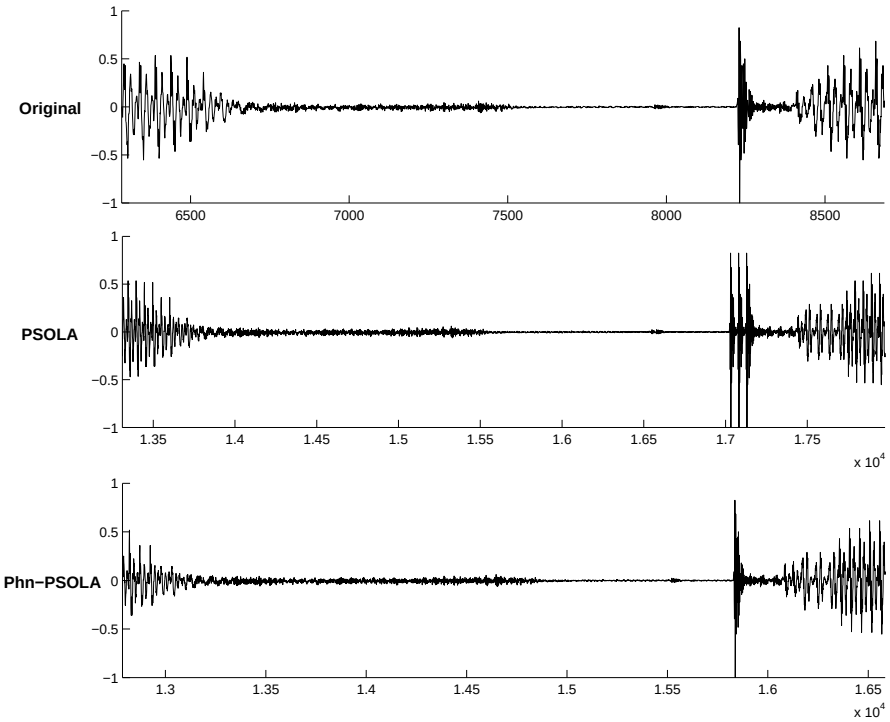
Vliv γ na celkovou změnu délky

Jednotlivé fonémy nejsou v češtině v souvislé řeči využívány rovnoměrně. Při jednom experimentu bylo v souboru slov zaznamenaných z mluvených veřejných projevů (zkoumaný soubor měl 187000 fonémů) napočítáno 41,3% vokálů a 58,7% konsonantů. [5] Tento výsledek je celkem pochopitelný, jelikož česká slova se skládají ze slabik, jež jsou většinou vytvářeny konsonanty ve spojení s vokály anebo slabikotvornými *r, l*. Část slabik je více než dvou písmená a k jednomu vokálu se tak připojí i dva konsonanty. Navíc *r, l* již patří mezi konsonanty, a některé slabiky se tak úplně obejdou bez vokálů. Výsledkem je tedy převaha konsonantů.

Důsledkem takovéto převahy je však to, že v řeči je též poměrně velké množství méně flexibilních fonémům. Takovéto fonémy pak způsobí, že signál požadované nové délky nedosáhne. Pro její dosažení je tak nutné zadat do algoritmu větší změnu délky - β. Výraznější změna více změní flexibilní fonémy – ty v součtu s méně flexibilními, které si udržují svoji původní délku, dohromady dosáhnou požadovanou délku signálu při přijatelné srozumitelnosti. Cenou za srozumitelnost je však i více či méně znatelná změna dynamiky hovoru.

5.2.3 Výsledný signál

Výsledek našeho snažení o zachování kvalit konsonantů nejlépe prezentuje obrázek 5.3.

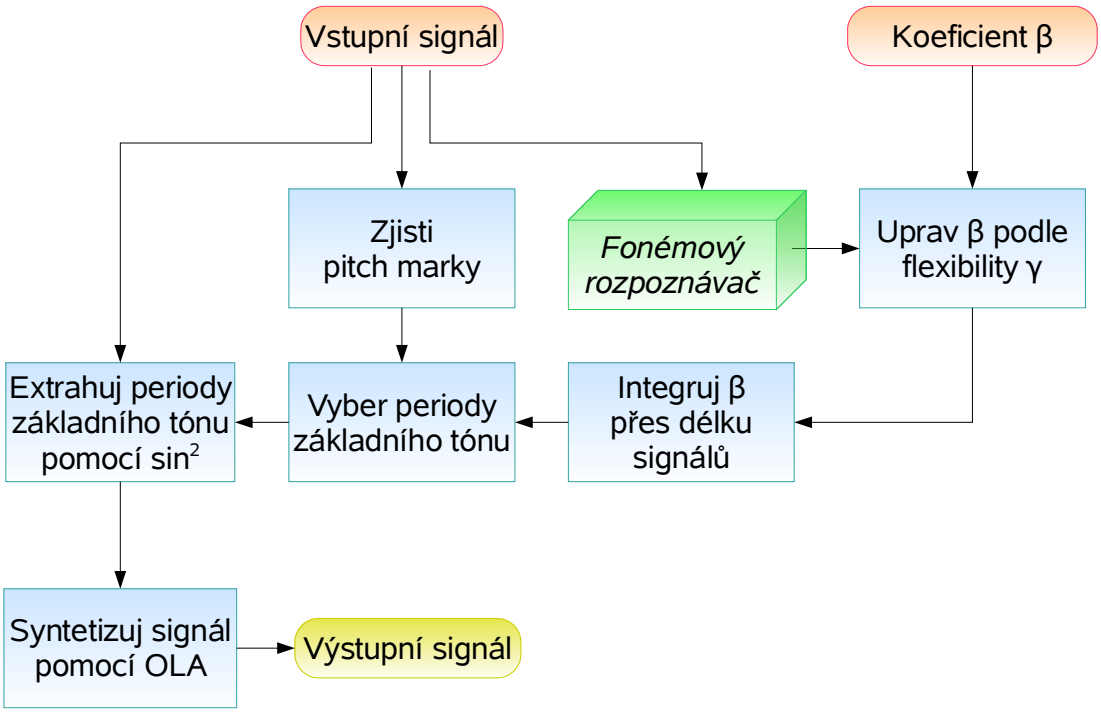


Obrázek 5.3: Úsek ze slova české po změně rychlosti na 0,5.

V horní části obrázku je průběh originálního signálu s fonémy *e, s , k, e:*. V druhém řádku je stejná část signálu ovšem se sníženou rychlostí (tedy signál je prodloužený) pomocí prosté PSOLy. Na třetím řádku je snížení rychlosti provedeno pomocí PSOLy s fonémovým rozšířením. V případě prosté PSOLy se jasně ukazuje vliv konstatní změny délky signálu na fonému *k*. Signál byl prodloužen a protože se při takové operaci periody základního tónu opakují, byl zopakován i úsek s tímto fonémem. Rozšířená PSOLA, ale ví, že oblasti fonému *k* má nízkou flexibilitu, a tudíž periody základního tónu neopakuje anebo jen minimálně.

Kapitola 6

Přehled kompletního systému



Obrázek 6.1: Celkový přehled navrženého systému.

Před tím, než přistoupím k popisu implementace, shrnu zde ve stručnosti, jak je celý systém navržen. Pomůckou mi k tomu bude blokové schéma na obrázku 6.1. Výklad zde bude na poměrně vysoké abstrakci.

Systém má dva vstupy – samotný signál a požadovaná změna jeho délky. Prvním krokem ve zpracování signálu je zjištění základního tónu v průběhu celého signálu a odvození poloh pitch marků. Tuto část realizuje některý z algoritmů HPS, Trasovač spojitého základního tónu či jakýkoliv jiný, jmenované dva však byly v projektu implementovány.

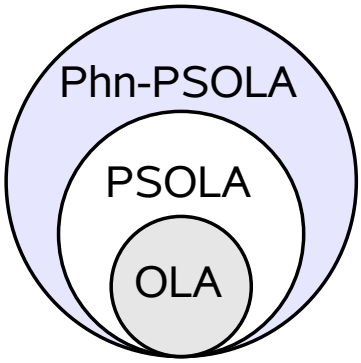
V dalším kroku je nutné určit fonémy v řečovém signálu, což provádí fonémový rozpoznávač. Jeho implementace nebyla součástí této práce a ani jeho vnitřní funkce není důležitá. Důležitý je jeho výstup, kterým je řada detekovaných fonémů nesoucí informace

o názvu fonému a poloze v čase v signálu. Tyto informace pak poslouží pro vyhledání flexibility. Systém dále již nepotřebuje znát jaký foném byl vysloven, stačí pouze hodnota flexibility v dané oblasti signálu. Pomocí flexibility je následně přepočítán koefiecient β.

Hodnota upravené β se zintegruje přes délku signálu. Délka integrálu je tedy stejná jako délka signálu. Každý vzorek má teď i svou hodnotu integrálu β. Dalším krokem je nyní určení period základního tónu, resp. pitch marky, které se projeví ve výsledném signálu. Tento krok se provevede pomocí informací o pozicích pitch marků v původním signálu a integrálu β.

S pomocí okénkovací funkce sin^2 a s využitím informací o pitch marcích se z původního signálu vyextrahují příslušené periody základního tónu. Segmenty se v závěru nad sebe patřičně nasunou a sečtou, čímž vznikne výstupní signál.

Z uvedeného diagramu je možné vyčíst jistou evoluci metody OLA, nad kterou je vystaven zbytek. OLA pouze sčítá překryté segmenty. To co z ní dělá PSOLu jsou bloky, které zjistí pitch marky a patřičné pitch periody vyberou okénkovací funkcí. Z pohledu OLy, tak ale opět dostává jen „nějaké“ segmenty, schodou okolností zarovnané na počátky period základního tónu. Posledním evolučním krokem je doplnění fonémového rozpoznávače, který periodám základního tónu dodá i obsah a napoví PSOLe jak s nimi zacházet. Tento nový krok bych nazval Phn-PSOLA.



Obrázek 6.2: Vývoj metody OLA

Kapitola 7

Ukázková aplikace

Významnou část mé práce tvoří demonstrační aplikace, která využívá metodu PSOLA a její rozšíření o fonémový rozpoznávač. Implementace PSOLy byla provedena tak, aby se maximálně usnadnil její přenos do jiných aplikací. Ukázková aplikace se tak dá vnímat jako jakýsi referenční návrh jak k modulu PSOLy přistupovat. Samotná aplikace má podobu konzolového programu. Program, jenž jsem nazval pplayer (Psola Player), provádí úpravu rychlosti řeči buď audio souboru ve formátu raw anebo živého vstupu ze zvukové karty. Soubory raw musí být monofonní se vzorky v 16 bitovém formátu. Při použití základní PSOLy mohou mít libovolnou vzorkovací frekvenci, při zapojení fonémového rozšíření však musí mít vzorkovací frekvenci pouze 8000 Hz, což je dáno použitým fonémovým rozpoznávačem z balíku BSAPI. Pro úplnost a porovnání jsem do aplikace doplnil i algoritmus *prokládání řečových rámců*. Ten je ovšem aktivní pouze pro úpravu souborů a nemá fonémové rozšíření.

PPlayer používá jedinou nestandardní knihovnu – BSAPI. Úspěšná kompilace tedy vyžaduje mít BSAPI ve funkčním stavu v adresáři BSAPI. Tato knihovna je závislá na dalších knihovnách pro optimalizované matematické výpočty, a proto je nutné tyto knihovny přilinkovat k mému programu. Více o kompilaci je v souboru readme.txt, který je přiložen u zdrojových kódů.

7.1 Struktura programu

Projekt je až na drobné vyjímky vypracován v jazyce C++. Pro návrh aplikace jsem využil jazyk UML, sloužícího pro návrh tříd, jejich interakcí apod. Jelikož mým vývojovým prostředím byl Linux, našel jsem aplikaci Umbrello. Umbrello umí vygenerovat zdrojové soubory z digramu tříd, čehož jsem s výhodou využil. Později v průběhu implementace jednotlivých funkčních celků jsem tento nástroj odložil, ale v závěru jsem znovu využil jeho schopnosti načíst hlavičkové soubory a vygenerovat z nich UML diagram tříd včetně jejich atributů a metod. Na obrázku 7.1 je zjednodušený diagram tříd mé aplikace.

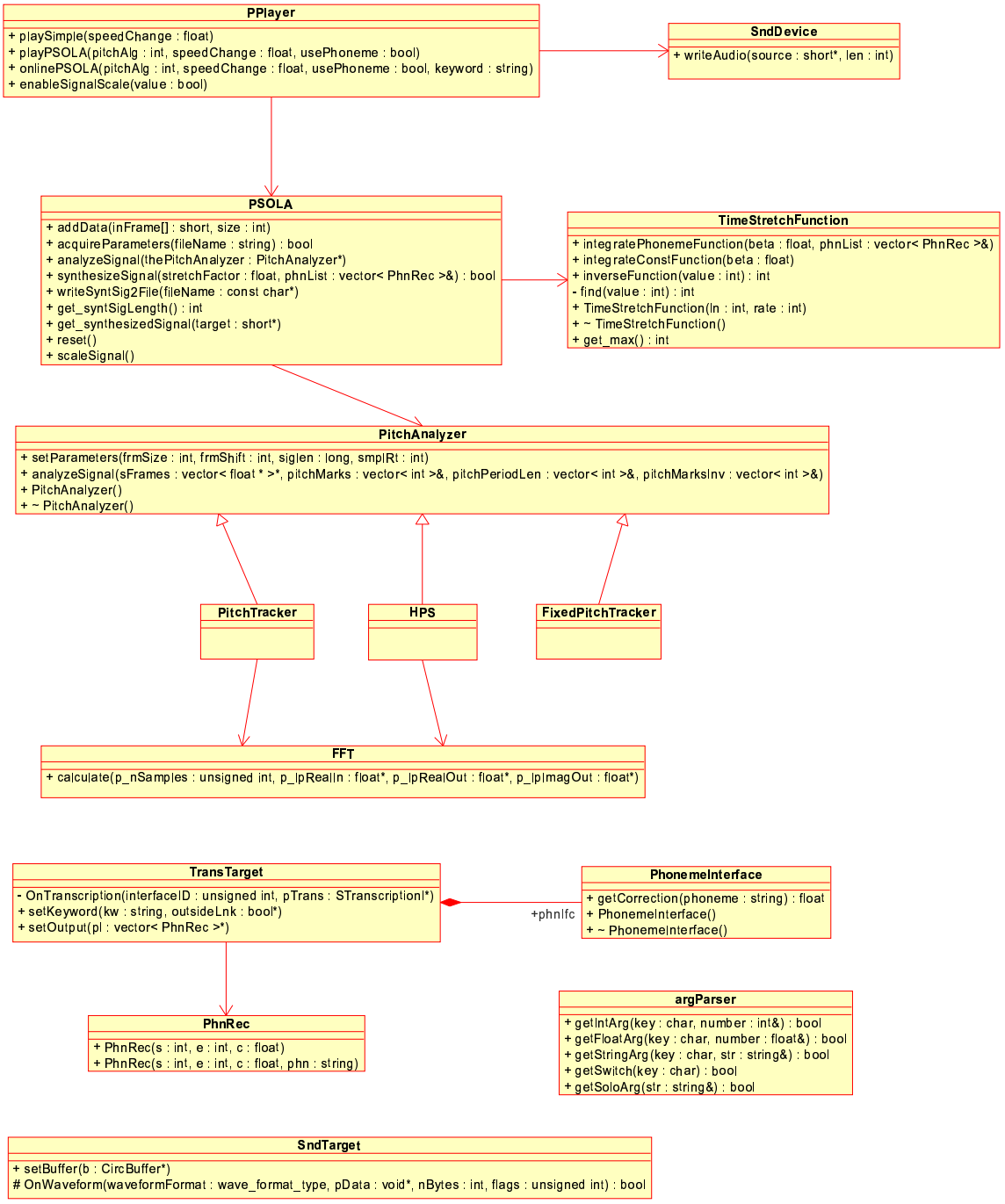
Dříve než shrnu jednotlivé třídy popíši části implentované v jazyce C.

V rámci snahy o urchlení výpočtů jsem realizoval funkce *sin* a *cos* pomocí vyhledávací tabulky. Tyto dvě funkce se nazývají `lut_sin` a `lut_cos`. Před jejich prvním použitím je nutné inicializovat vyhledávací tabulku funkcí `lutInit`.

Další sada funkcí je několik variant `dlmwrite`, které slouží pro uložení jednorozměrných dat – polí a vektorů. Tyto funkce vytvářejí soubor se stejným formátováním jako funkce dlmwrite v matalbu a matlab je tak dokáže načíst funkcí dlmread. Dlmwrite slouží především pro vývoj, kdy je často potřeba zkontrolovat mezivýsledky zpracování signálu, a do výsled-

ného spustitelného souboru se vkládají pouze je-li definováno makro `DEBUG`.

Nyní tedy stručný popis tříd v projektu. Pro jednoduchost budu metody v popisu uvádět bez parametrů.



Obrázek 7.1: Diagram tříd ukázkové aplikace

ArgParser slouží k extrakci jednotlivých parametrů předávaných programu při spuštění z konzole. Umí rozpoznat číselné a řetězcové argumenty uvedené klíčovým písmenem

ve tvaru '-x', přepínače ve tvaru '-x' a samostatný řetězcový parametr bez úvodního klíče (slouží např. jako jméno vstupního souboru).

PPlayer zabaluje jádro aplikace, které provádí načítání dat, a využívá objekt třídy `PSOLA` ke jejich zpracování. PPlayer obsahuje tři veřejné metody, které spustí samotný proces zpracování souboru – jednou je `playSimple`, která implementuje změnu rychlostí algoritmem *prokládání řečových rámců*, druhou je metoda `playPSOLA`, která pro změnu rychlosti využije *PSOLu* a třetí je metoda `onlinePSOLA`, která skrze BSAPI načítá ze zvukové karty data, které ukládá do kruhového bufferu a následně je z něj vybírá a zpracovává pomocí PSOLy. Čtvrtou veřejnou metodou je enableSignalScale, která umožní skrze třídu PSOLA upravit hodnoty vzorků signálu tak, aby využívaly celého rozsahu <-1,1>. Metody playPSOLA a onlinePSOLA rovněž umožňují aktivovat fonémové rozšíření v objektu PSOLA. Do objektu PSOLA už vstupuje pouze seznam nalezených fonémů. Komunikaci s fonémovým rozpoznávačem zde obstarává právě objekt třídy PPlayer. PSOLA je tak nezávislá na použitém fonémovém rozpoznávači a je možné jej jednoduše měnit.

PSOLA je třída, která plně implementuje metodu PSOLA. Její rozhraní umožňuje vkládat data libovolné délky (`addData`). PSOLA získává potřebné parametry signálu (pitch marks atd.) v metodě (`analyzeSignal`), kde využívá objektů odvozených z třídy PitchAnalyzer, při volání metody analyzeSignal se jí takový objekt předá jako parametr. Kromě zjištění parametrů za běhu programu je možné je i načíst předpočítané z externích souborů (`acquireParameters`). AcquireParameters rozumí formátu souboru vytvořeného funkcí dlmwrite v matlabu. Po načtení požadovaných dat se tato zpracují algoritmem PSOLA pro požadovanou změnu rychlosti, respektive poměr nové délky vůči původní. K tomuto kroku slouží metoda `synthesizeSignal`. SynthesizeSignal má dva parametry – stretchFactor značící koeficient β a phnList, což je vektor objektů třídy PhnRec, tedy seznam nalezených fonémů. V případě, že je tento vektor neprázdný, aktivuje se fonémové rozšíření PSOLy. Výsledný řečový signál je možné uložit do souboru (`writeSyntSig2File`) anebo do předem připraveného pole (`get_synthesizedSignal`). Pole je nutné alokovat ručně a k určení, jak velké pole se má vytvořit, slouží metoda `get_syntSigLength`. Třída PSOLA má i další metody, jejich výčet a popis je v dokumentaci v adresáři `documentation` na přiloženém CD.

TimeStretchFunction se využívá objektem PSOLA při generování nové posloupnosti pitch marků. Implementuje integraci koeficientu β. Třída obsahuje dvě metody pro samotnou integraci – `integrateConstFunction` a `integratePhonemeFunction`. Obě přijímají parametr β a fonémová varianta navíc vektor nalezených fonémů. Důležitou metodou je také `inverseFunction`, která zjišťuje, který vzorek nese nějakou konkrétní hodnotu integrálu.

PitchAnalyzer je základní třída pro tzv. pitch markery - algoritmy, které v signálu určují pozici pitch marků a další potřebné údaje. Z této třídy se nevytvářejí objekty, pouze slouží jako rodičovská třída, která svým potomkům definuje rozhraní, kterým jsou dvě metody – `setParameters`, kterou se do pitch markeru předají parametry signálu jako je délka signálu, vzorkovací kmitočet aj., a `analyzeSignal`, která provede samotnou analýzu signálu a určí pozice pitch marků.

PitchTracker je potomkem PitchAnalyzeru a implementuje algoritmus Smooth Pitch Tracker. Při vytváření instance této třídy se určí frekvenční pásmo, ve kterém se

má základní tón pohybovat. Dále je možné určit, jestli se mají vypisovat informace o průběhu analýzy, která probíhá ve třech krocích – výpočet chyb zkoušených základních tónů, výpočet nejlepší cesty skrze matici chyb z předchozího kroku a posledním krokem je z cesty určit přesnou pozici pitch marků.

FixedPitchTracker je potomkem PitchAnalyzeru. Signál však nijak neanalyzuje, pouze generuje řadu pitch marků v pevných odstupech. V pplayeru je zvolen odstup odpovídající základnímu tónu 125 Hz. Použitím tohoto algoritmu se PSOLA degraduje na ekvivalent algoritmu OLA.

HPS je potomkem PitchAnalyzeru a implementuje algoritmus Harmonic Product Spectrum. Při vytváření instance této třídy se určí frekvenční pásmo, ve kterém se má základní tón pohybovat. Dále je možné určit, jestli se mají vypisovat informace o průběhu analýzy, která probíhá ve třech krocích – součin podvzorkovaných spekter, výpočet nejlepší cesty skrze matici součinů z předchozího kroku a posledním krokem je z cesty určit přesnou pozici pitch marků.

PhnRec tato struktura udržuje informace o nalezených fonémech – název fonému, jeho počátek a konec v signálu a flexibilitu neboli γ.

PhonemeInterface se využívá pro určení flexibility fonému. Tvoří rozhraní k souboru phoneme.def, kde jsou tyto hodnoty uloženy.

CircBuffer implementuje kruhový zásobník, který slouží pro krátkodobé uložení hlasových dat, pro uvažovanou aplikaci, která by při detekci klíčového slova v projevu přehrála posledních několik sekund rychlejším tempem a dostihla tak „reálný čas“.

FFT obsahuje implementaci rychlé Fourierovy transformace.

SndDevice je jednoduchá třída umožňující přehrávání zvuku použitím rozhraní OSS (Open Sound System).

TransTarget poskytuje call-back metodu pro fonémový rozpoznávač z BSAPI. Zpracovává nalezené fonémy a pomocí PhonemeInterface jim přiřazuje flexibilitu.

SndTarget poskytuje call-back metodu pro záznam zvuku. V ukázkové aplikaci plní kruhový buffer zvukovými daty.

7.2 Funkce

Využití PSOLy je poměrně jednoduché a přímočaré. Metody jsou pojmenovány tak aby byly v maximální míře sebepopisné. Po vytvoření objetu třídy PSOLA tak zpracování signálu sestává z postupného zavolání čtyř metod:

```
PSOLA thePSOLA(sampleRate);
vector<PhnRec> phnList;

thePSOLA.addData(inputData, size);

if(usePhoneme){
  // Analýza signálu fonémovým rozpoznávačem z BSAPI
```

```
  //  - naplnění vektoru phnList fonémy
}

thePSOLA.analyzeSignal(&PitchTracker(50, 350));
thePSOLA.synthesizeSignal(Beta, phnList);
thePSOLA.writeSyntSig2File(fileName);
```

V tomto příkladu byl jako pitch marker použit Trasovač spojitého základního tónu s rozsahem 50 – 350 Hz. Po analýze signálu může být metoda synthesizeSignal volána bez omezení s různými hodnotami β.

7.3 Ovládání

Aplikace přijímá při spouštění řadu parametrů. Zde je jejich úplný seznam. Při špatném vstupu je na obrazovku tento seznam vypsán.

```
Usage:
  pplayer -r <sampling rate> -s <speed> <source file> [other options]

Options:
 -b buffer capacity in seconds (integer). Default 5 seconds
 -m method
    0 - simple
    1 - PSOLA - direct file (default)
    2 - PSOLA - audio recording through buffer
    When option 2 is selected, no source file is needed.
 -p switch enabling phoneme recognition
 -a pitch mark algorithm
    1 - Smooth Pitch Tracker (default)
    2 - Fixed-Pitch tracker (degrades PSOLA to OLA)
    3 - Harmonic Product Spectrum
 -o output file. Default out.raw.
 -c enable signal scaling - may improve pitchmark detection,
    but increases volume
 -k keyword
```

Při spuštění programu na souboru se na obrazovku vypíše krátké shrnutí zvolených parametrů a v závislosti na vybraném algoritmu detekce základního tónu také průběh této analýzy. Je-li povoleno fonémové rozšíření, vypíší se na obrazovku ještě informace o inicializaci této komponenty.

Je-li program spuštěn s volbou živého vstupu, zobrazí se opět krátký souhrn parametrů a začne záznam zvuku do kruhového bufferu o zvolené délce. Program nyní čeká na stisk klávesy enter, po jejímž stiknutí začne odebírat zvuk z bufferu, měnit jeho rychlost a současně přehrávat. Po dobu přehrávání bufferu stále probíhá záznam a buffer se průběžně doplňuje. Smysluplné je tedy zvuk z bufferu přehrávat pouze zvýšenou rychlostí. V současnosti je však problém s rychlostí pokročilých algoritmů na detekci základního tónu. Na mém počítači s procesorem Intel Pentium M 1,5 Ghz s 512 MB RAM totiž nedosahují reálného času, a tak se buffer nikdy nevyprázdní. Jediným dostatečně rychlým algoritmem

se zdá být *Fixed-Pitch tracker*. Není-li použito fonémové rozšíření, tak je rychlý dostatečně, je-li však fonémový rozpoznávač aktivní, přesáhne součet zatížení mez reálného času a zvuk není plynulý.

Myšlenkou na pozadí této části programu je vytvořit aplikaci, která „poslouchá“ zvukový zdroj a zaznamenává jej do kruhového bufferu. Poslouchánim je myšleno zpracování signálu fonémovým rozpoznávačem. V okamžiku, kdyby „zaslechl“ klíčové slovo, začal by se přehrávat zvuk z bufferu, tedy o několik sekund za reálným časem, zvýšenou rychlostí, až by dohnal reálný čas a v tom okamžiku by se na něj přepnul. Tato část zůstává pouze rozpracována, jelikož fonémový rozpoznávač ještě nemá dokončenou část na tzv. keyword-spotting – z tohoto důvodu je aktivace přehrávání závislá na stisku klávesy enter, který klíčové slovo simuluje.

Kapitola 8

Zhodnocení implementovaných metod

Hodnocení řečového projevu s pohledu přirozenosti a srozumitelnosti není možné provést automatizovaně pomocí prostředků výpočetní techniky. Testování tak muselo proběhnout se skutečnými posluchači. Vybral jsem 2 posluchače, kteří měli za úkol ohodnotit upravené zvukové záznamy. Samotné hodnocení sestávalo z určení srozumitelnosti a čistoty řečového signálu.

Srozumitelnost není velký problém při snížení rychlosti, spíše než srozumitelnost se tehdy hodnotí čistota hlasu. Při zvýšené rychlosti je čistota hlasu též dúležitým faktorem, významu zde ovšem nabývá srozumitelnost. Testování probíhalo následujícím způsobem.

Jak jsem vysvětlil v sekci 5.2.2, délka souboru upraveného pomocí fonémově rozšířené PSOLy závisí na fonémové skladbě řeči. Čím více závěrových konsonantů obsahuje, tím se pro danou změnu rychlosti od sebe odlišuje výsledná délka souboru PSOLy a Phn-PSOLy. Pomocí Pplayeru jsem tedy vygeneroval řadu souborů s různým zrychlením (až do rychlosti 4,5) s variantou PSOLA a Phn-PSOLA. Poté jsem našel velikostně přibližně shodné páry vždy jeden PSOLA a druhý Phn-PSOLA. Posluchač se pak vyjádřil o srozumitelnosti a porovnal tyto dvě ukázky.

Při testování změny rychlosti řeči není možné obecně prohlásit, že nějaká metoda umožní s dobrou srozumitelností zrychlit signál na konkrétní úroveň. Každý člověk totiž mluví různým tempem a fonémy tak mají při záznamu určenu svoji délku. Hovoří-li človek rychle a tento záznam se pak pokusíme zrychlit, je zřejmé, že k vysokým hodnotám zrychlení nedospějeme. Opakem je pomalý hovor, který je možné urychlit poměrně značně. Z uvedeného plyne, že hodnoty zrychlení, které ještě posluchač považuje za srozumitelné, mají vypovídající hodnotu pouze v kontextu konkrétního zvukového vzorku. Větší hodnotu bude mít komparativní hodnocení obyčejné PSOLy a PSOLY s fonémovým rozšířením.

Testování probíhalo na zvukových vzorcích ze sady *spdat* s délkou přibližně jedné věty. Vzorky byly dvou typů:

- Mužský hlas – 8000 Hz, 16b
- Ženský hlas – 8000 Hz, 16b

Další testy byly provedeny s vyšší vzorkovací frekvencí, a tudíž pouze pro samotnou PSOLu. Testovány byly krátké záznamy z televize ČT24.

Výsledky

Obecně lze říci, že v kvalitě řeči metoda PSOLA jako taková (ať už prostá nebo fonémová) jasně předčila prokládání řečových rámců. Samozřejmě při zvolení Fixed-Pitch trackeru jako pitch markeru, kdy se PSOLA degraduje na OLu, byla standardní PSOLA také bezpochyby lepší.

Kvalita řeči se mění podle toho, jakou operaci se signálem provádíme – zrychluje-li se řeč, je kvalita subjektivně lepší než při zpomalení. Zpomalená řeč začíná zvýrazňovat některé ruchy v signálu. Například ve vzorcích spdat lze při pozorném poslouchání zaznamenat ozvěnu z místnosti, kde byly nahrávány. Po zpomalení je toto echo znatelnější.

Vzorek	Metoda	
	PSOLA	***Phn-PSOLA***
a30000s7 mužský hlas	**3** Mluvčí jakoby spěchal	**4,5** Lepší srozumitelnost, ale dynamika řeči je ovlivněna
	2,5 Mluvčí spěchá	**3,5** Lepší srozumitelnost, klidnější hlas
	2 Mluvčí spěchá	**2,5** Lepší srozumitelnost, klidnější hlas
a30000o5 mužský hlas	**3** Mluvčí jakoby spěchal	**4** Klidnější hlas
	2,5 Mluvčí spěchá	**3,5** Klidnější hlas
	1,75	**2** Téměř shodné, mírně klidnější hlas
a30018s9 ženský hlas	**3** Lepší srozumitelnost	**4,5** Špatná dynamika brání srozumitelnosti
	2,5	**3,5** Trochu lepší srozumitelnost
	1,75	**2** Prakticky shodné
a30018s6 ženský hlas	**3** Lepší srozumitelnost	**4,5** Špatná dynamika, část nesrozumitelná
	2,5	**3,5** Poněkud horší dynamika, klidnější projev
	2 Mluvčí spěchá	**2,5** Klidnější hlas

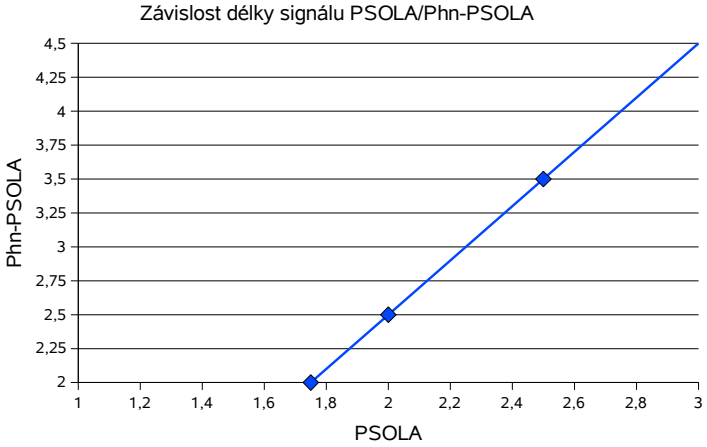
Tabulka 8.1: Ukázka výsledků porovnání. Čísla značí změnu rychlosti. Porovnávány jsou v řádcích výsledky s podobnou délkou výstupního signálu.

Z uvedené tabulky je patrné, že lepší výsledky podává Phn-PSOLA v případě mužského hlasu. Hlas zní klidněji a hlásky jsou jakoby pečlivěji artikulovány. U vyšších zrychlení již však dochází k tomu, že se citelněji mění poměry mezi více a méně flexibilními fonémy, což má v krajním případě negativní vliv na srozumitelnost. V takových případech má poněkud lepší srozumitelnost prostá PSOLA, protože ačkoli jsou některé krátké fonémy potlačeny (např. k) mozek si je dokáže takříkajíc domyslet v rámci dynamiky na kterou je zvyklý.

Pozoruhodné jsou horší výsledky u ženského hlasu. Původní signál měl přibližně stejnou rychlost hovoru jako v případě mužských vzorků, zrychlený signál však často trpěl horší dynamikou proslovu, což ovlivňovalo i srozumitelnost, která je často lepší u prosté PSOLy.

Důvodem je zřejmě ne zcela přesné určení fonémů fonémovým rozpoznávačem. Jeho výstup pro dané vzorky vykazoval poměrně dost nepřesností. Vliv by mohl mít i vyšší základní tón ženského hlasu.

Dalším zjištěním jsou opakující se dvojice zrychlení, které si navzájem odpovídají výslednou délkou signálu. Vyneseme-li nejčastější dvojice zrychlení PSOLA – Phn-PSOLA (1,75 – 2; 2 – 2,5; 2,5 – 3,5; 3,0 – 4,5) do grafu (obrázek 8.1), dojdeme k zajímavému zjištění, že leží na přímce. Tato lineární závislost je zřejmě způsobena podobným rozložením tuhých a pružných fonémů v testovaných vzorcích.



Obrázek 8.1: Závislost délky výstupního signálu PSOLA – Phn-PSOLA

V případě snížení rychlosti byl v obou variantách PSOLy zvuk poměrně čistý. Phn-PSOLA si však lépe poradila se závěrovými hláskami, které zněly čistěji v porovnání s variantou PSOLA. Při té dík opakování segmentů tyto hlásky „drnčely“ – namísto jedné exploze např. u hlásky „p“ se jich vygenerovalo několik za sebou.

Při poslechových testech upravených nahrávek z ČT24 s vyšším vzorkovacím kmitočtem se potvrdila celková vysoká úroveň PSOLy. Až na ojedinělé zvukové artefakty způsobené nepřesným určením pitch marku byl řečový signál zcela čistý. Použitelná rychlost se nicméně pohybovala na hranici 2,5 násobku originální rychlosti, zde by zřejmě mohla pomoci Phn-PSOLA, která však díky limitaci fonémového rozpoznávače nebyla testovatelná.

Pro úplnost ještě dodám hodnocení algoritmu prokládání řečových rámců. Při jeho testování se projevovaly zvukové artefakty, způsobené navazováním rámců, které byly vytrženy v různých fázích základního tónu a jednoduše spojeny jeden za druhým bez prolnutí. Ve výstupním signálu se ozývalo praskání a hlas sám zněl drsně. Přes uvedené problémy však bylo bez problému možné určit mluvčího. Na základě testování se jako ještě srozumitelná jevila dvojnásobná rychlost. Při vyšších násobcích již byla řeč natolik poznamenána výpadky hlásek, že bylo obtížné rozumět slovům obzvláště slyšel-li je posluchač poprvé. Při aplikaci zpomalení řeči byl hlas sice stále srozumitelný, ale zněl velmi chraptivě.

Jedinou výhodou tohoto algoritmu je rychlost, jež se dá srovnat s přímým kopírováním souborů, jelikož audio signál neprochází žádnou úpravou, kromě rozdělení na rámce.

Kapitola 9

Závěr

V této diplomové práci jsem splnil všechny body jejího zadání. V písemné části jsem rozebral principy tvorby řečového signálu a z toho odvodil postupy nutné pro změnu rychlosti řeči resp. změnu délky signálu. V přehledu metod pro změnu rychlosti řeči byla uvedena metoda fázového vokodéru pracující ve frekvenční oblasti a především metoda PSOLA pracující v časové oblasti. Kromě těchto algoritmů, které jsou v praxi používány, jsem vytvořil i jednoduchý algoritmus pro změnu rychlosti, který jsem nazval Prokládání řečových rámců.

Algoritmus PSOLA jsem implementoval v jazyce C++. Modul PSOLA je znovupoužitelný i v jiných aplikacích. Jeho implementace umožňuje snadnou výměnu algoritmu pro určení pitch marků. Implementovány byly algoritmy Trasovač spojitého základního tónu a Harmonic Product Spectrum. Preferovaným z těchto dvou je Trasovač spojitého základního tónu, jelikož nejpřesněji určuje pozice pitch marků. HPS není stabilní a především mužské hlasy s nižším základním tónem neurčuje správně, je však výrazně rychlejší než Trasovač spojitého základního tónu.

Řečový signál se skládá z fonémů a jejich charakter je poměrně různorodý, byla tedy definována nová vlastnost fonému – **flexibilita** (γ), určující to, jak je foném „pružný“. Pro algoritmus PSOLA bylo vypracováno rozšíření, které upravuje změnu délky pro jednotlivé fonémy české fonémové sady dle jejich flexibility. Výsledkem tohoto rozšíření je lepší srozumitelnost řeči především při zvýšení její rychlosti. Návrh a realizace tohoto systému jsou nezávislé na zpracovávaném jazyce. Pro jiný jazyk by se pouze natrénoval fonémový rozpoznávač na novou sadu fonémů, kterým by se také určila jejich flexiblita. I integrace samotného fonémového rozpoznávače umožňuje jeho snadnou výměnu.

S využitím modulu PSOLA byla vytvořena demonstrační aplikace, která upravuje zvukové soubory a živý audio vstup. Výstupy z této aplikace potvrzují platnost teorií přednesených v této práci, tedy že PSOLA s fonémovým rozšířením ve většině případů zlepšuje srozumitelnost. Praktické se jeví maximálně 2,5 – 3 násobné zrychlení v závislosti na výchozím signálu.

9.1 Možný budoucí rozvoj

- Aby PSOLA dostála svému jménu – tedy Pitch Synchronous OLA, musí mít zjištěny co nejpřesněji pitch marky. Současná implementace Pitch Trackeru (sekce 4.1.3) je sice velmi přesná, ale také velmi pomalá jelikož intenzivně využívá trigonometrické funkce. Nedá se u ní mluvit o real-timu. Další implementovaný algoritmus HPS je rychlý – bohužel však na úkor přesnosti. Návrh třídy PSOLA umožňuje vcelku snadné

začlenění nových algoritmů, takže jedním z možných směrů, jakým by se projekt mohl dále rozvíjet je výkonnější pitch marker. Cestou k výkonnějšímu pitch markeru by mohla být optimalizace stávajících dvou algoritmů anebo vytvoření nového. Např. implementace HPS je poměrně jednoduchá, případné vyladění tohoto algoritmu by mohlo přinést použitelnější výsledky.

- Dalším rozšířením, které by nemělo být příliš komplikované, je umožnit využívání fonémového rozpoznávače natrénovaného na 8000 Hz i na signály s vyšší vzorkovací frekvenci pomocí on-line podvzorkování na 8 kHz. Rozpoznávač podává na frekvenci 8000 Hz dobré výsledky a analýza signálu s vyšším vzorkovacím kmitočtem by byla výpočetně náročnější, přičemž výrazně lepší výsledky by zřejmě nepřinesla.

- Dělení fonémů je z pohledu flexibility poměrně hrubé, fonémový rozpoznávač by se tedy mohl natrénovat tak, aby rozpoznával jen čtyři skupiny fonémů – vokály, úžinové souhlásky a závěrové souhlásky a polozávěrové. Aktuální verze fonémového rozpoznávače podává při jeho současné sadě fonémů dobré výsledky, avšak při zjednodušení dělení fonémů by se mohlo dosáhnout ještě větší přesnosti v rámci nového dělení.

- Zlepšení dynamiky řeči výsledného signálu by mohlo také prospět, kdyby fonémový rozpoznávač přesněji určoval hranice fonémů v signálu. Časté jsou totiž hranice zasahující do sousedních fonémů. V případě, kdy je vedle sebe tuhý a pružný foném, např. *te*, a hranice je určena až ve fonému *e*, je jeho část v dalším zpracování vnímana jako méně flexibilní. Při změně rychlosti signálu pak nedojde k jejímu adekvátnímu zkrácení příp. prodloužení a dynamika je tím ovlivněna.

Literatura

[1] Dutoit, T.: *An Introduction to Text-to-Speech Synthesis*. Kluwer academic Publishers, 1997, ISBN 0-7923-4498-7.

[2] Holzapfel, M.; Hoffmann, R.; Höge, H.: A Wavelet-Domain PSOLA Approach. Technická zpráva, Siemens Corp. & Institute for Technical Acoustics, Technical University of Dresden.

[3] Hála, B.; Sovák, M.: *Hlas řeč sluch:Základy fonetiky a logopedie*. Státní pedagogické nakladatelství, 1962.

[4] Kolektiv autorů: Audio timescale-pitch modification. Navštíveno: 10. 12. 2006. URL http://en.wikipedia.org/wiki/Audio_timescale-pitch_modification

[5] Krčmová, M.: *Fonetika a fonologie; Zvuková stavba současné čestiny*. Masarykova Universita, Fakulta filozofická, 1990.

[6] Lemmetty, S.: *Review of Speech Synthesis Technology*. Diplomová práce, Helsinki University of Technology, March 1999.
URL http://www.acoustics.hut.fi/publications/files/theses/lemmetty_mst/thesis.pdf

[7] Middleton, G.: Pitch Detection Algorithms. Navštíveno: 1. 10. 2006.
URL http://cnx.org/content/m11714/latest/

[8] Sethares, W. A.: A Phase Vocoder in Matlab. Navštíveno: 8. 12. 2006.
URL http://eceserv0.ece.wisc.edu/~sethares/vocoders/phasevocoder.html

[9] Sigmund, M.: *Analýza řečových signálů: přednášky*. VUT v Brně – Fakulta Elektrotechniky a Informatiky.

[10] Szöke, I.: Smooth Pitch Tracker Based on Harmonic and Noise Model. In *STUDENT EEICT 2005*, Faculty of Information Technology BUT, 2005, ISBN 80-214-2890-2, s. 673–677.
URL http://www.fit.vutbr.cz/research/view_pub.php?id=7777

[11] Sündermann, D.; Bonafonte, A.; et. al.: Frequency Domain vs. Time Domain VTLN. Technická zpráva, Universitat Politècnica de Catalunya (UPC), Department of Signal Theory and Communications, 2004.

[12] Černocký, J.: *Číslicové zpracování řeči – přednášky*. VUT v Brně – Fakulta Informačních Technologií.

Dodatek A

Sada fonémů použitá v projektu

Foném	Význam	Foném	Význam
a	a	m	m
a:	á	N	měkkopatrové n (η)
a_u	au	n	n
b	b	o	o
c	ť	o:	ó
d	d	o_u	ou
d_Z	dž	P_	ř
d_z	dz	p	p
e	e	pau	pauza
e:	é	r	r
e_u	eu	S	š
F	potlačené m (μ)	s	s
f	f	spk	ruch pozadí
g	g	t	t
h_	h	t_S	č
i	i	t_s	c
i:	í	u	u
int	ruch pozadí	u:	ú
J	ň	v	v
J_	ď	x	ch
j	j	Z	ž
k	k	z	z
l	l	oth	ruch pozadí

Tabulka A.1: Seznam českých fonémů používaných v BSAPI

Tabulka A.2 obsahuje flexibilty fonémů (γ) použité při výpočtu změny délky signálů. Tato tabulka reflektuje obsah souboru *phoneme.def*.

Foném	γ	Foném	γ
a	1,0	m	0,3
a:	1,0	N	0,3
a_u	1,0	n	0,3
b	0,3	o	1,0
c	0,3	o:	1,0
d	0,3	o_u	1,0
d_Z	0,7	P_	1,0
d_z	0,7	p	0,3
e	1,0	pau	1,0
e:	1,0	r	1,0
e_u	1,0	S	1,0
F	0,7	s	1,0
f	1,0	spk	1,0
g	0,3	t	0,3
h_	1,0	t_S	0,3
i	1,0	t_s	0,3
i:	1,0	u	1,0
int	0,8	u:	1,0
J	0,3	v	1,0
J_	0,3	x	1,0
j	1,0	Z	1,0
k	0,3	z	1,0
l	1,0	oth	1,0

Tabulka A.2: Flexibility fonémů